Lecture Notes in Computer Science 7645

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Li Xu Elisa Bertino Yi Mu (Eds.)

Network and System Security

6th International Conference, NSS 2012
Wuyishan, Fujian, China, November 21-23, 2012
Proceedings

Springer

Volume Editors

Li Xu
Fujian Normal University
School of Mathematics and Computer Science
Fuzhou, Fujian 350108, China
E-mail: xuli@fjnu.edu.cn

Elisa Bertino
Purdue University
Department of Computer Science, CERIAS and Cyber Center
West Lafayette, IN 47907-2107, USA
E-mail: bertino@purdue.edu

Yi Mu
University of Wollongong
School of Computer Science and Software Engineering
Wollongong, NSW 2522, Australia
E-mail: ymu@uow.edu.au

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-34600-2 e-ISBN 978-3-642-34601-9
DOI 10.1007/978-3-642-34601-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012950605

CR Subject Classification (1998): K.6.5, C.2.0-1, E.3, H.2.7, D.4.6, K.4.4, C.2.4-5, C.4

LNCS Sublibrary: SL 4 – Security and Cryptology

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

NSS 2012, the 6th International Conference on Network and System Security, was held in Wuyishan, Fujian, China, during November 21–23, 2012. The conference was organized and supported by the School of Mathematics and Computer Science, Fujian Normal University, China.

NSS is a series of events covering research on all theoretical and practical aspects related to network and system security. The aim of NSS is to provide a leading edge forum to foster interaction between researchers and developers within the network and system security communities, and to give attendees an opportunity to interact with experts in academia, industry, and government.

In response to the call for papers, 173 papers were submitted to NSS 2012. These papers were evaluated on the basis of their significance, novelty, technical quality, and practical impact. The review and discussion were held electronically using EasyChair. Of the papers submitted, 39 were selected for inclusion in this Springer volume (LNCS 7645), giving an acceptance rate lower than 23%.

The conference also featured three keynote speeches, by Rajkumar Buyya entitled "Market-Oriented and Energy-Efficient Cloud Computing," by Ravi Sandhu entitled "The Future of Access Control: Attributes, Automation and Adaptation," and by Wanlei Zhou entitled "Traceback of Distributed Denial-of-Service (DDoS) Attacks," respectively.

We are very grateful to the people whose work ensured a smooth organization process: Xinyi Huang, Muttukrishnan Rajarajan, and Yong Guan, Publicity Chairs, for their work in ensuring the wide distribution of the call for papers and participation; Shengyuan Zhang and Ayong Ye, Organizing Chairs, for taking care of the local organization; and Xiaohui Hu for managing the conference website.

Last but certainly not least our thanks go to all authors who submitted papers and all attendees. We hope you enjoy the conference proceedings!

November 2012

Li Xu
Elisa Bertino
Yi Mu

Organization

General Co-chairs

Qidan Ling	Fujian Normal University, China
Peter Mueller	IBM Zurich Research

Program Co-chairs

Li Xu	Fujian Normal University, China
Elisa Bertino	Purdue University, USA
Yi Mu	University of Wollongong, Australia

Steering Chair

Yang Xiang	Deakin University, Australia

Workshop Co-chairs

Avinash Srinivasan	Bloomsburg University, USA
Eric Pardede	Latrobe University, Australia
Shui Yu	Deakin University, Australia

Organization Chairs

Shengyuan Zhang	Fujian Normal University, China
Ayong Ye	Fujian Normal University, China

Publicity Chairs

Xinyi Huang	Fujian Normal University, China
Muttukrishnan Rajarajan	City University London, UK
Yong Guan	Iowa State University, USA

Program Committee

Rafael Accorsi	University of Freiburg, Germany
Gail-Joon Ahn	Arizona State University, USA
Eric Alata	LAAS-CNRS, France
Joonsang Baek	Khalifa University of Science, Technology and Research, UAE

Marina Blanton	University of Notre Dame, USA
Carlo Blundo	Università di Salerno, Italy
Barbara Carminati	University of Insubria, Italy
Marco Casassa Mont	Hewlett-Packard Labs, UK
David Chadwick	University of Kent, UK
Tingting Chen	Oklahoma State University, USA
Xiaofeng Chen	Xidian University, China
Zhide Chen	Fujian Normal University, China
Jung Hee Cheon	Seoul National University, Korea
Mauro Conti	University of Padua, Italy
Jorge Cuellar	Siemens Corporate Technology, Germany
Frédéric Cuppens	Telecom Bretagne, France
Robert H. Deng	Singapore Management University, China
Roberto Di Pietro	Università di Roma Tre, Italy
Ning Ding	Shanghai Jiao Tong University, China
Xuhua Ding	Singapore Management University, Singapore
Wenliang Du	Syracuse University, USA
Elena Ferrari	University of Insubria, Italy
Simone Fischer-Huebner	Karlstad University, Sweden
Keith Frikken	Miami University, USA
Steven Furnell	University of Plymouth, UK
Alban Gabillon	University of French Polynesia, France
Joaquin Garcia-Alfaro	Telecom Bretagne, France
Gabriel Ghinita	University of Massachusetts, Boston, USA
Dieter Gollmann	Hamburg University of Technology, Germany
Juan Gonzalez Nieto	Queensland University of Technology, Australia
Lein Harn	University of Missouri-Kansas City, USA
Jiankun Hu	UNSW@ADFA, Australia
Qiong Huang	South China Agricultural University, China
Michael Huth	Imperial College London, UK
Limin Jia	Carnegie Mellon University, USA
Audun Josang	University of Oslo, Norway
James Joshi	University of Pittsburgh, USA
Murat Kantarcioglu	University of Texas at Dallas, USA
Guenter Karjoth	IBM Research-Zurich
Sokratis Katsikas	University of Piraeus, Greece
Stefan Katzenbeisser	TU Darmstadt, Germany
Muhammad Khurram Khan	King Saud University, Saudi Arabia
Shinsaku Kiyomoto	KDDI Laboratories Inc., Japan
Costas Lambrinoudakis	University of Piraeus, Greece
Bo Lang	Beihang University, China
Adam J. Lee	University of Pittsburgh, USA
Laurent Lefevre	INRIA, France
Hui Li	Xidian University, China
Yingjiu Li	Singapore Management University, Singapore

Yong Li	Beijing Jiaotong University, China
Changlu Lin	Fujian Normal University, China
Peng Liu	The Pennsylvania State University, USA
Shengli Liu	Shanghai Jiao Tong University, China
Giovanni Livraga	Università degli Studi di Milano, Italy
Der-Chyuan Lou	Chang Gung University, Taiwan
Li Lu	University of Electronic Science and Technology of China, China
Jianfeng Ma	Xidian University, China
Fabio Martinelli	IIT-CNR, Italy
Carlos Maziero	Federal University of Technology - Paraná, Brazil
Qun Ni	Google Inc.
Eiji Okamoto	University of Tsukuba, Japan
Jaehong Park	University of Texas at San Antonio, USA
Jong Hyuk Park	Kyungnam University, Korea
Gerardo Pelosi	Politecnico di Milano, Italy
Günther Pernul	Universität Regensburg, Germany
Alexander Pretschner	Karlsruhe Institute of Technology (KIT), Germany
Indrakshi Ray	Colorado State University, USA
Ning Shang	Qualcomm, USA
Elaine Shi	University of Maryland, College Park, USA
Harry Skianis	University of the Aegean, Greece
Miguel Soriano	Universitat Politècnica de Catalunya (UPC), Spain
Anna Squicciarini	The Pennsylvania State University, USA
Hung-Min Sun	National Tsing Hua University, Taiwan
Willy Susilo	University of Wollongong, Australia
Qiang Tang	University of Luxembourg, Luxembourg
Juan E. Tapiador	Universidad Carlos III de Madrid, Spain
Dan Thomsen	Smart Information Flow Technologies, USA
Mahesh Tripunitara	The University of Waterloo, Canada
Traian Marius Truta	Northern Kentucky University, USA
Jaideep Vaidya	Rutgers University, USA
Zhiguo Wan	Tsinghua University, China
Guilin Wang	University of Wollongong, Australia
Huaxiong Wang	Nanyang Technological University, Singapore
Jian Wang	Nanjing University of Aeronautics and Astronautics, China
Leon Wang	National University of Kaohsiung, Taiwan
Lingyu Wang	Concordia University, Canada
Duncan S. Wong	City University of Hong Kong, Hong Kong
Qianhong Wu	Rovira i Virgili University, Tarragona, Spain
Xiaoxin Wu	Huawei Security Research Lab, China

Shouhuai Xu	University of Texas at San Antonio, USA
Chung-Huang Yang	National Kaohsiung Normal University, Taiwan
Xun Yi	Victoria University, Australia
Kangbin Yim	Soonchunhyang University, Korea
Fangguo Zhang	Sun Yat-sen University, China
Rui Zhang	Institute of Information Engineering, Chinese Academy of Sciences, China
Yuqing Zhang	Graduate University of Chinese Academy of Sciences, China
Zhenfeng Zhang	Institute of Information Engineering, Chinese Academy of Sciences, China
Jianying Zhou	Institute for Infocomm Research, Singapore
Liehuang Zhu	Beijing Institute of Technology, China
Zutao Zhu	Google Inc.

External Reviewers

Aafer, Yousra	He, Kun	Ratazzi, Paul
Al Khalil, Firas	Hong, Hyunsook	Sgandurra, Daniele
Aliasgari, Mehrdad	Iskander, Marian	Shi, Jie
Au, Man Ho	Islam, Mohammad	Suarez-Tangil, Guillermo
Baracaldo, Nathalie	Jacobson, David	Sun, Shifeng
Barenghi, Alessandro	Jayaraman, Karthick	Sun, Yue
Ben Jaballah, Wafa	Ke, Pinhui	Sun, Yuhua
Broser, Christian	Khadilkar, Vaibhav	Taghavi Zargar, Saman
Cai, Shaoying	Kim, Jinsu	Takabi, Hassan
Canlar, Eyup	Kim, Myungsun	Tan, Xiao
Chan, Aldar C-F.	Kim, Taechan	Ulusoy, Huseyin
Chen, Kai	Krautsevich, Leanid	Vrakas, Nikos
Darra, Eleni	Lai, Junzuo	Vu, Quang Hieu
Deng, Hua	Lazouski, Aliaksandr	Wang, Yi
Diouri, Mohammed El Mehdi	Lee, Chin-Feng	Wang, Yifei
	Li, Jin	Wang, Yujue
Drogkaris, Prokopios	Liang, Kaitai	Xu, Jia
Gao, Wei	Lin, Hsiao-Ying	Yang, Li
Garcia-Alfaro, Joaquin	Long, Xuelian	Yoon, Eunjung
Geneiatakis, Dimitris	Ma, Xu	Zhang, Lei
Gonzalez, Jesus	Maganis, Gabriel	Zhang, Tao
Gu, Haihua	Meier, Stefan	Zhang, Weijie
Guarino, Stefano	Niu, Yuan	Zhang, Xin
Guglielmi, Michele	Ntantogian, Christoforos	Zhang, Zijian
Guo, Xu	Paulet, Russell	Zhao, Bin
Guo, Zheng	Peter, Andreas	Zhao, Jun
Hassan, Sabri	Rachapalli, Jyothsna	

Table of Contents

Network Security I

System Security I

Public Key Cryptography I

Privacy I

Authentication I

Network Security II

System Security II

Public Key Cryptography II

Security Analysis

Privacy II

Authentication II

Access Control

Enhancing List-Based Packet Filter Using IP Verification Mechanism against IP Spoofing Attack in Network Intrusion Detection

Yuxin Meng and Lam-for Kwok

Department of Computer Science
City University of Hong Kong
Hong Kong SAR, China
ymeng8@student.cityu.edu.hk

Abstract. Signature-based network intrusion detection systems (NIDSs) have become an essential part in current network security infrastructure to identify different kinds of network attacks. However, signature matching is a big suffering problem for these systems in which the cost of the signature matching is at least linear to the size of an input string. To mitigate this issue, we have developed a context-aware packet filter by means of the blacklist technique to filter out network packets for a signature-based NIDS and achieved good results. But the effect of the whitelist technique has not been explored in our previous work. In this paper, we therefore aim to develop a list-based packet filter by combining the whitelist technique with the blacklist-based packet filter under some specific conditions, and investigate the effect of the whitelist on packet filtration. To protect both the blacklist and the whitelist, we employ an IP verification mechanism to defend against IP spoofing attack. We implemented the list-based packet filter in a network environment and evaluated it with two distinct datasets, the experimental results show that by deploying with the IP verification mechanism, the whitelist technique can improve the packet filtration without lowering network security.

Keywords: Intrusion Detection System, Network Packet Filter, List Technique, Network Security and Performance, IP Verification.

1 Introduction

Network threats (e.g., virus, worms, phishing sites) have become a big problem to current network communications. To mitigate this issue, network intrusion detection systems (NIDSs) [1,2] have been widely deployed in personal computers and public networks with the purpose of defending against different kinds of network attacks.

In general, network intrusion detection systems can be classified into two categories: *signature-based NIDS* and *anomaly-based NIDS*. The signature-based NIDS [5] (also called *misuse-based NIDS* or *rule-based NIDS*) detects an attack by comparing its signatures with incoming packet payloads. The *signature* is a kind of descriptions for a known attack. On the other hand, the anomaly-based NIDS [6,7] detects an attack by discovering significant deviations between its established normal profile and observed

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 1–14, 2012.

events. A *normal profile* represents the normal behavior of a user or a network collection. In real deployment, the method of comparing signatures is more popular than the approach of identifying anomalies, the reason is that the signature-based NIDS can relatively achieve lower false alarm rate [11].[1]

Problem. Although NIDSs have become an important and essential part to current network security infrastructure, it is a big challenge for these intrusion detection systems, especially for the signature-based NIDS in a high volume traffic environment in which they could drop a large number of network packets due to the expensive signature matching procedure. For example, Snort [3,8] which is an open-source signature-based NIDS, can quickly exhaust a computer's memory in a heavy traffic environment and discard huge amounts of packets since the number of network packets from the intensive web-traffic environment can greatly exceed its maximum processing capability [4]. The terrible performance of a signature-based NIDS under such environment could cause lots of security issues (i.e., missing some deleterious packets may leave out network attacks). The consuming time of a signature-based NIDS is mainly spent in comparing their signatures with incoming packet payloads in which the computing consumption is at least linear to the size of an input string [9].

To mitigate the above issue, we have previously proposed and developed a context-aware blacklist-based packet filter [22], which was based on blacklist technique to reduce the burden of a signature-based NIDS by pre-filtering out a number of network packets. In particular, we generated the blacklist by means of a statistic-based method (called *weighted ratio-based blacklist generation*) in the component of *monitor engine* to calculate the IP confidence. In the experiment, we initially evaluated the context-aware blacklist-based packet filter with the DARPA dataset and the experimental results showed that our approach encouragingly achieved an improvement over the processing time than the traditional signature-based intrusion detection systems (e.g., Snort).

Motivation. The list-technique, which consists of both a blacklist and a whitelist, has been extensively investigated in the field of spam detection. In our previous work [22], we presented a novel work by constructing a network packet filter with the blacklist technique. In real settings, we notice that the blacklist technique is more prevalent than the whitelist technique since the whitelist is considered too costly to network security. For instance, a network packet can pass through a NIDS directly if this packet's source IP address is recorded in the whitelist. However, we find that by providing with some certain protections and under some specific conditions, the whitelist may be very useful in real applications. For instance, the whitelist technique has been widely used to protect a network server from DDoS attack [15,18]. In this work, our motivation is therefore to construct a list-based packet filter with an IP verification mechanism and investigate the effect of the whitelist technique on packet filtration.

Contributions. In our previous work [22], we have not investigated the effect of the whitelist technique on packet filtration. With the above motivation, in this paper, we therefore attempt to combine the whitelist technique with the previously developed blacklist-based packet filter, and evaluate the effect of the whitelist technique on refining

[1] False alarm rate (FAR) is the total number of false negatives and false positives divided by the total number of alarms.

network packets. Specifically, to defend against the IP spoofing attack which is the major challenge for the list-based packet filter, we employed an IP verification mechanism behind the blacklist and the whitelist to safeguard network security. In the experiment, we deployed the developed *context-aware list-based packet filter* into a network environment, evaluating with the DARPA dateset and a real dataset respectively. The experimental results show that by protecting with the IP verification mechanism, the list-based packet filter can effectively prevent the IP spoofing attack, and the whitelist technique can offer further improvements on packet filtration without lowering network security. The contributions of our work can be summarized as below:

- We developed a *context-aware list-based packet filter* by integrating the whitelist technique into our previously developed blacklist-based packet filter. Being aware of the limitations of the whitelist, we conducted the packet filtration by using the whitelist under some specific conditions.
- We identify that IP spoofing is a major attack for a list-based packet filter. To defend against this attack, we employ an IP verification mechanism to protect both the blacklist and the whitelist. By deploying with this mechanism, the list-based packet filter can achieve a good packet filtration rate and filtration accuracy without lowering network security.
- In the experiment, we evaluated the list-based packet filter with both the DARPA dataset and a real dataset. The experimental results show that the whitelist technique can make a positive impact on packet filtration and that, by protecting with the IP verification mechanism, the list-based packet filter can achieve a better outcome.

The remaining parts of this paper are organized as follows: we introduce our previously developed context-aware blacklist-based packet filter and describe some related work regarding to the applications of whitelist technique in Section 2; in Section 3, we describe the architecture of the context-aware list-based packet filter and detail the interactions among its components; we describe the experimental methodology and results in Section 4; at last, we conclude our work in Section 5.

2 Background

In this section, we begin by briefly introducing our previously developed context-aware blacklist-based packet filter and then we describe some related work about the applications of whitelist technique.

2.1 Context-Aware Blacklist-Based Packet Filter

In our previous work [22], we proposed and developed a *context-aware blacklist-based packet filter* to reduce the processing burden of a signature-based NIDS by pre-filtering out massive network packets. The previous analysis also showed that this filter would not lower network security even under an IP spoofing attack. The architecture of deploying the context-aware blacklist-based packet filter is shown in Fig. 1.

In real settings, a NIDS is usually deployed in front of an internal network so as to protect the whole network environment from network attacks by monitoring the network traffic and producing alert messages. Therefore, the *context-aware blacklist-based*

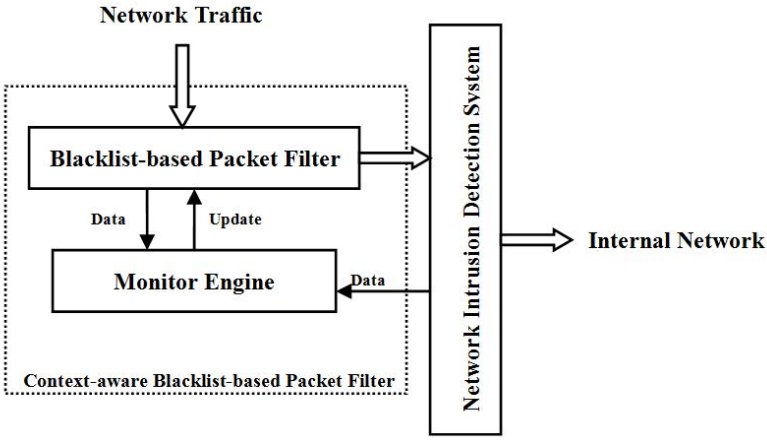


Fig. 1. The architecture of deploying the context-aware blacklist-based packet filter in network intrusion detection

packet filter is implemented in front of the NIDS to alleviate its burden by filtering out a large number of network packets. As shown in Fig. 1, there are mainly two developed components in the *context-aware blacklist-based packet filter*: a *blacklist-based packet filter* and a *monitor engine*. The *blacklist-based packet filter* is the core component that filters out network packets and compares packet payloads with NIDS signatures in terms of their IP addresses. The *monitor engine* is responsible for calculating the IP confidence by collecting the network data (e.g., alarms) transmitted from both the NIDS and the *blacklist-based packet filter*. In addition, the *monitor engine* will periodically update the blacklist in the *blacklist-based packet filter* by using a method of *weighted ratio-based blacklist generation*. The formula of the *weighted ratio-based blacklist generation* is given as below:

$$\textit{IP confidence} = \frac{\sum_{i=1}^{n} i}{\sum_{k=1}^{m} 10 \times k} (n, m \in \mathbf{N})$$

In the formula, i represents the number of good packets, k represents the number of bad packets and 10 is the provided weighted value. More specific details of the blacklist generation and the component interactions can be refer to our previous work [22].

In practice, network traffic will first reach the *blacklist-based packet filter* and the filter will search for IP addresses in the blacklist. If the IP address of the packet is in the blacklist, then the filter will further compare its payload with the NIDS signatures. If a match is identified, then the blacklist-based packet filter will block this packet and generate an alert. At the same time, a short message will be sent out from the *blacklist-based packet filter* to the *monitor engine*. On the other hand, if the packet payload does not match any NIDS signatures, then this packet will be sent to the Internal Network.

In addition, if the IP address of the packet is not found in the blacklist, then the packet will be sent to the NIDS for examination. The alert messages generated from the NIDS will be collected by the monitor engine in calculating IP confidence. Through

interacting with the NIDS and comparing the packet payloads with the NIDS signatures, the *context-aware blacklist-based packet filter* can filter out network packets without affecting network security even under an IP spoofing attack.

2.2 Related Work

The whitelist technique, which is used to describe a list of entities that are authorized with a particular privilege, has been widely deployed and studied in the field of spam detection. The whitelist can be used to identify the senders from who users are willing to accept emails [14] and can be implemented into a challenge-response system as an anti-spam solution that is able to accompany with currently existing anti-spoofing techniques [10].

Currently, the whitelist technique are also being applied into many other fields. For example, Lofgren and Hopper [12] employed the whitelist in the anonymous authentication and presented a system of FAUST, a TTP-Free scheme that eliminated the blacklist and replaced it with an implicit whitelist. The system of FAUST therefore could allow an anonymous user to prove that he or she was authorized to access an online service, however, if the user misbehaved, he or she could retain the anonymity but would be unable to authenticate in future sessions.

In addition, a lot of studies utilize the whitelist technique to defend against several network attacks such as *DDoS* attack. Kim *et al.* [20] proposed a whitelist-based defense scheme that increased connection success ratio of legitimate connection requests under SYN flooding attacks. Their experimental results showed that their proposed method could ensure a high connection success ratio for legitimate clients under the SYN flooding. Then, Chen and Itoh [18] proposed a whitelist-based approach to defend against flooding attacks on a SIP server. This approach was capable of keeping the most comprehensive and up-to-date information about the legitimate SIP clients. However, it has some limitations in dealing with attacks from a botnet. To mitigate this problem, Yoon [15] presented a new survival strategy by using the whitelist technique. This work proposed that the IP addresses, which were from the previously successful logins, should be gathered to make a whitelist (called very important IP addresses-*VIP list*). When the victim is under a DDoS attack, they give higher priority to the traffic that belongs to the *VIP list*. Their analysis and experimental results showed that the *VIP list* could enable the Critical Internet Sites (CIS) to continue its business even under a severe DDoS attack scenario.

In this work, we mainly attempt to construct a list-based packet filter by implementing the whitelist technique into our developed context-aware blacklist-based packet filter under some conditions, and to investigate the effect of the whitelist on packet filtration, which has not been explored in our previous work. In terms of the previous studies, we identify that IP spoofing is a major attack for the list-based packet filter. Therefore, to protect the blacklist and the whitelist against IP spoofing attack, we further employ an IP verification mechanism behind the list when filtering out network packets. The evaluation presents positive results by adding the whitelist technique in the aspect of packet filtration. Additionally, the results also indicate that by deploying with our developed IP verification mechanism, the list-based packet filter can efficiently filter out network packets without lowering network security.

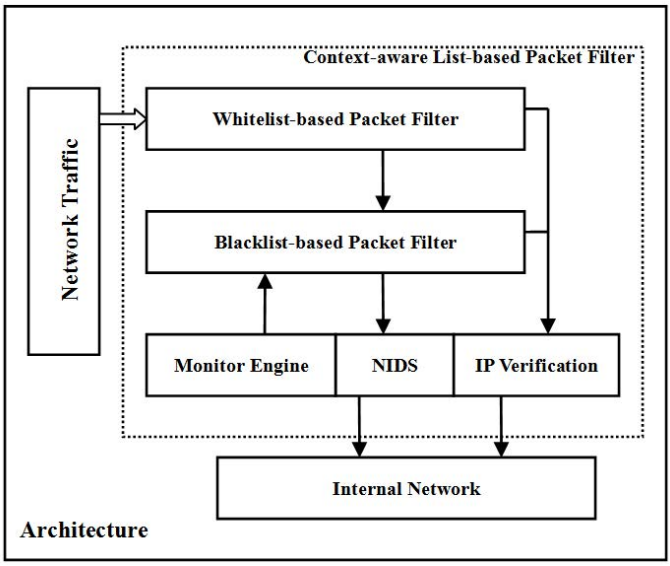


Fig. 2. The high-level architecture of deploying the context-aware list-based packet filter in network intrusion detection environment

3 Context-Aware List-Based Packet Filter

The *list technique* consists of both a blacklist and a whitelist. By combining the whitelist technique with the developed blacklist-based packet filter, we accordingly name the new one as the *list-based packet filter*, since the new packet filter contains both the blacklist and the whitelist. The high-level architecture of constructing and deploying the *context-aware list-based packet filter* is shown in Fig. 2.

In order to combine the whitelist with the blacklist-based packet filter, we mainly develop a component of *whitelist-based packet filter* as shown in Fig. 2, and we deploy it in front of the blacklist-based packet filter. Therefore, there are totally five major components in the *context-aware list-based packet filter*: *a whitelist-based packet filter*, *a blacklist-based packet filter*, *a monitor engine*, *an IP verification module* and *a network intrusion detection system (NIDS)*.

In the remaining parts, we first describe the construction of the whitelist-based packet filter. Then, we present the interactions among the whitelist-based packet filter, the blacklist-based packet filter, the monitor engine, the IP verification module and the network intrusion detection system. Finally, we describe the IP verification module how to protect the list-based packet filter against IP spoofing.

3.1 Whitelist-Based Packet Filter

The construction of the whitelist-based packet filter is illustrated in Fig. 3. There are mainly two parts: *Whitelist* and *Look-up Table*. The *Whitelist* contains all whitelisting IP addresses while the *Look-up Table* is optional which contains two sub-tables:

Whitelist	**Look-up Table**	
Source IP address	*Special Conditions*	*General Conditions*
IP address1	**None**	**Flag Information, Type of Service, Destination Port Number, etc.**
IP address2	**Source Port Number…**	
IP address3	**Total Length**	
IP address4	**None**	
……	……	

Fig. 3. The construction of the whitelist-based packet filter: *Whitelist* and *Look-up Table*

Special Conditions and *General Conditions*. The table of *General Conditions* stores all additional conditions that can be compared with incoming packets such as Flag information, Type of service, etc. The table of *Special Conditions* records some special conditions for a particular source IP address. Take the *source port number* as an example, this port number could be extracted from the historic data that has been proven to be free from attacks. In real settings, the two tables are mainly constructed by using expert knowledge (i.e., experts' advice and analysis) and analyzing the historic data.

If the two tables contain any conditions, then the incoming packets will be allowed to the Internal Network if and only if these packets meet all the conditions. Two possible situations of the packet filtration in the *whitelist-based packet filter* are described as below:

- If a packet is from *IP address1* or *IP address4*, then this packet will be directly compared with the conditions in the table of *General Conditions* since there is no content in the table of *Special Conditions*.
  - If all conditions in the table of *Special Conditions* are matched, then the packet will be directly sent to the *IP Verification Module*.
  - If at least one condition is not matched, then the packet will be sent to the *blacklist-based packet filter*.
- If a packet is from *IP address2* or *IP address3*, then this packet will be first compared with the conditions (e.g., Flag information, Total length) in the table of *Special Conditions*.
  - If all conditions are matched, then the packet will be compared with the conditions in the table of *General Conditions*.
  - If not all conditions are matched in any of these two tables, then the packet will be sent to the *blacklist-based packet filter*.
  - If and only if all conditions in the two tables are matched, the packet can be sent to the *IP Verification Module*.

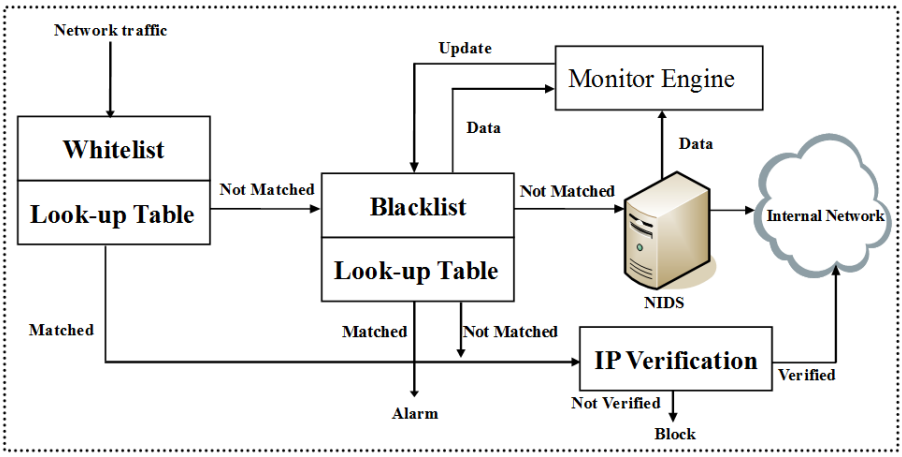


Fig. 4. The interactions among the whitelist-based packet filter, the blacklist-based packet filter, the monitor engine, the IP verification module and the network intrusion detection system

3.2 Interactions among Components

The explicit interactions among the components in the *context-aware list-based packet filter* are illustrated in Fig. 4.

In real settings, when a packet arrives, it will reach the *whitelist-based packet filter* at first. If the IP address of this packet is in the whitelist, then this packet will be compared with all the conditions in the look-up table.

- If all the conditions are matched, then the packet will be sent to the *IP verification module*.
  - If the IP verification process is successful, then the packet can be sent to *Internal Network* directly without the examination from the NIDS.
  - If the IP verification process is not successful, then the packet will be blocked.
- If at least one condition is not matched, then the packet will be sent to the *blacklist-based packet filter* and NIDS.

On the other hand, if the IP address of the packet is not in the whitelist, then the packet will be sent to the *blacklist-based packet filter*. The checking procedure in the *blacklist-based packet filter* is described as below:

- If the IP address of the packet is in the blacklist, then the payload of this packet will be compared with the signatures in its look-up table.[2]
  - If a match is found, then the filter will block this packet and make an alarm. Besides, a message will be sent out from the *blacklist-based packet filter* to the *monitor engine*, reporting the generated alarm and relevant IP address.

[2] The look-up table in the *blacklist-based packet filter* is organized different from that in the *whitelist-based packet filter*. Details can be referred to our previous work [22].

  - If the payload of the packet does not match any of the signatures, then this packet will be sent to the *IP verification module*. If and only if the IP verification is successful, the packet can be allowed to enter into *Internal Network*.
- If the IP address is not in the blacklist, then the packet will be sent to the NIDS, and the NIDS will examine the packet in the traditional way.

After the examination, the NIDS will report the packet state (e.g., good or bad) of the IP address to the monitor engine. Finally, the monitor engine calculates the IP confidence based on collected data and updates the blacklist periodically.

3.3 IP Verification Module

Based on our previous work, we identify that the IP-reputation based *list-technique* including both the whitelist and the blacklist is vulnerable to IP spoofing attack. To mitigate this problem, in this work, we deploy an *IP verification module* behind both the whitelist-based packet filter and the blacklist-based packet filter to defend against the IP spoofing attack by verifying IP sources (see Fig. 4).

The IP verification is a method of verifying whether an agent truly represents the service or the user it claims to represent. Several commercial IP verification systems (e.g., IP Source Guard [16], Synopsys Verification IP [17]) have been used in anti-DDoS systems to defend against IP spoofing attacks. The IP verification procedure can be conducted at a network layer, a transport layer, or an application layer according to the demand.

In general, the verification procedure is that, when a *connection request packet* arrives, the IP verification system sends back a *responding packet* to the client address. If a valid response comes back from the client, then the IP verification system will allow the packets to the target network. Otherwise, the IP verification system will block this packet and record this event.

Specifically, there are many ways to realize the IP verification such as *traceback techniques* [23,24] and *prevention techniques* [25,26]. As a proof of concept, in this work, we adopted the *prevention technique* and developed the IP verification module by using and modifying an open-source java-based project.[3] This project is a P2P program so that we need to install it on both communication hosts. In particular, we denoted the *program* deployed in the source host as *client module* and only denoted the *program* deployed in the destination host (namely the *filter*) as *IP verification module*. The detailed IP verification procedure and steps are described as below.

- When a packet arrives, the *IP verification module* sends back a UDP packet with a randomly generated secret N to the *client module* of the source IP address.
- The *client module* receives the UDP packet and the secret N. Then, the *client module* sends back a UDP packet with a secret $N + 1$.
- The *IP verification module* receives the UDP packet and the secret $N + 1$. The task is to verify whether the secret value is correct.
  - If the secret value is correct, then the IP source is regarded to be true.
  - If the secret value is false or the *IP verification module* does not receive the UDP packet, then the IP source is regarded as forged source.

[3] *Rodi:* `http://rodi.sourceforge.net/wiki/`

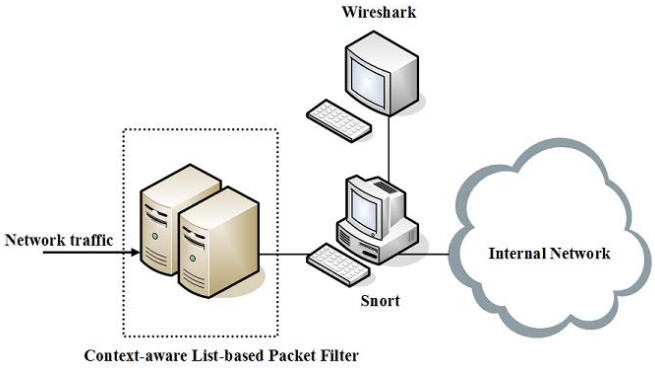


Fig. 5. The experimental network deployment including the context-aware list-based packet filter, Snort and Wireshark

4 Evaluation

In this section, we explore the performance of the *context-aware list-based packet filter* and investigate the effect of the whitelist on packet filtration under some conditions in an experimental environment constructed with Snort [8] and Wireshark [13]. The experimental deployment is illustrated in Fig. 5.

In the experimental environment, we used the Snort version 2.9.0.5 with the default rule configuration. We implemented the context-aware list-based packet filter into a server and deployed it in front of Snort. In this case, network traffic has to first arrive at the list-based packet filter for packet filtration. The Wireshark is responsible for monitoring network traffic and assisting in calculating the IP confidence in the *blacklist-based packet filter*.

4.1 Experimental Methodology

In the evaluation, we mainly conducted two experiments (named *Experiment1* and *Experiment2*) to investigate the performance of the list-based packet filter. In the *Experiment1*, we used the DARPA dataset to explore the effect of the whitelist technique on packet filtration. In the *Experiment2*, we utilized a real dataset to investigate the performance of the list-based packet filter in filtering out network packets. The whitelist and its look-up table (namely *conditions*) are constructed by using expert knowledge (e.g., Honeypot-experts' analysis) and pre-analyzing the datasets.

- *Experiment1:* To explore the effect of the whitelist technique on packet filtration, we evaluated the list-based packet filter by using the DARPA dataset [21]. In our previous work, we also conducted the evaluation with this dataset. Therefore, by comparing the two results, we can identify the effect of the whitelist technique on packet filtration.
- *Experiment2:* In this experiment, we utilized a real dataset, which was collected by a Honeypot project, to explore the performance of the context-aware list-based

Table 1. Evaluation results of the consuming time under the two experimental situations

Situation1: *Consuming Time (s)*			**Situation2:** *Consuming Time (s)*		
Week Day	Week2	Week5	*Week Day*	Week2	Week5
Monday	14	13	Monday	11.3	9.07
Tuesday	16	17	Tuesday	12.1	14.5
Wednesday	5	16	Wednesday	5	11.4
Thursday	13	27	Thursday	9.4	20
Friday	11	38	Friday	8.3	28.6

Table 2. Results of the saving-time percentage for previous and current work

Saving time (%)	*Previous Work*		*Current Work*	
Week Day	*Week2*	*Week5*	*Week2*	*Week5*
Monday	14.3	23.1	19.3	30.2
Tuesday	18.8	11.8	24.4	14.8
Wednesday	0	18.8	0	23.1
Thursday	23.1	22.2	27.7	26.0
Friday	18.2	21.1	24.5	24.7

packet filter in filtering out network packets. Additionally, we also evaluated the blacklist-based packet filter with this real dataset, so that we can explore the effect of the whitelist by comparing the two results.

In the *Experiment1*, based on our previous work, we whitelisted 57 IP addresses that were free from any attacks in the DARPA dataset without *conditions*, while 123 IP addresses were in the whitelist for *Experiment2* with both *special conditions* and *general conditions* by pre-analyzing the real dataset. The *analysis work* for constructing the whitelist and the look-up table in the *Experiment2* was conducted by Honeyport experts through considering the Honeypot deployment and settings (i.e., provided services, opened port numbers), and analyzing, labeling the collected data. In addition, the settings regarding to the *context-aware blacklist-based packet filter* is the same as our previous work (i.e., using "1" as the assigned ratio threshold).

4.2 Evaluation with DARPA Dataset

Followed by the above experimental methodology and the experimental settings, we conducted the experiment with DARPA dataset in terms of two situations: (1) *Situation1:* without deploying the context-aware list-based packet filter; (2) *Situation2:* deploying the context-aware list-based packet filter. The results of the consuming time in the experiment are shown in Table 1.

Analysis: As shown in Table 1, we find that with the deployment of the context-aware list-based packet filter, the consuming time of Snort is decreased. To better illustrate the performance, we compare our current results with the previous work in Table 2. In the table, it is easily visible that by combining the whitelist technique, the list-based packet filter (namely the current work) further improves the percentage range of the saving-time from [11.8%, 23.1%] to [14.8%, 30.2%].

Table 3. Results of the saving-time percentage of the context-aware blacklist-based packet filter and the context-aware list-based packet filter compared to the performance of Snort

Day Time	*Context-aware Blacklist-based Packet Filter*	*Context-aware List-based Packet Filter*
DAY1	25.4%	30.1%
DAY2	30.7%	34.7%
DAY3	27.5%	29.8%

4.3 Evaluation with Real Dataset

We conducted another experiment by replacing the DARPA dataset with a real network traffic trace to explore the performance of the list-based packet filter. The real dataset contains 3-day data (e.g., DAY1, DAY2 and DAY3) which was collected by a Honeypot deployed in our CSLab. The Honeypot[4] opened several services (e.g., HTTP) and recorded all incoming traffic including both normal packets and malicious packets. The performance results of the two filters compared to Snort are shown in Table 3.

Analysis: In the table, we find that the *context-aware blacklist-based packet filter* can improve the performance of Snort in the range from 25.4% to 30.7%, while the saving-time percentage range of the *context-aware list-based packet filter* is from 29.8% to 34.7%. It is easily visible that the *context-aware list-based packet filter* can achieve a better result in packet filtration, and the results also indicate that the whitelist technique can make a positive impact on packet filtration by improving the filtration rate.

4.4 IP Spoofing Defend

IP spoofing is a big threat to the list-based packet filter, which can create an IP packet with a forged source IP address. To mitigate this attack, we deploy an IP verification mechanism behind the developed *context-aware list-based packet filter*.

In the same experimental environment, we conducted an initial verification that used a packet generator [19] to test this IP verification mechanism through generating 1000 forged packets and 100 valid packets. The Wireshark shows that the deployed *IP verification module* can identify all the forged IP packets and valid packets. The test results present that the *context-aware list-based packet filter* can filter out network packets without lowering network security. A larger test will be performed in our future work.

In addition, based on the experimental results, we provide two guidelines for generating the whitelist as the whitelist is so expensive to affect network security.

- The generation of the whitelist should be fully controlled by a security administrator, any other persons cannot modify the whitelist without the granted privilege. In addition, the security administrator should update the whitelist periodically to ensure its reliability and security.
- For an organization, reserved internal IP addresses that have no chance to launch an attack can be possibly recorded in the whitelist. These IP addresses should be examined and updated periodically.

[4] The Honeypot project is managed by HoneybirdHK `http://www.honeybird.hk/`

5 Concluding Remarks

The expensive signature matching process is a big problem for a signature-based NIDS in a large-scale network. To mitigate the issue, we previously developed a context-aware blacklist-based packet filter to help filter out a number of packets. However, the effect of the whitelist technique on packet filtration has not been investigated.

In this paper, we therefore focus on constructing a *context-aware list-based packet filter* by combining the whitelist technique with our previously developed blacklist-based packet filter under some conditions, and explore the performance of the whitelist technique on packet filtration. Moreover, we deploy an IP verification module behind both the whitelist and the blacklist to defend against IP spoofing attack. We implemented the context-aware list-based packet filter in a network environment and evaluated it with the DARPA dataset and a real dataset respectively. The experimental results show that the whitelist technique can further improve the performance of packet filtration. In addition, we conducted another experiment to verify the deployed IP verification mechanism and the results proved that this mechanism could safeguard the list-based packet filter against IP spoofing attack and guarantee the effective packet filtration without lowering network security.

Our work presents an early result of evaluating the performance of the context-aware list-based packet filter. The future work could include further exploring the performance of this packet filter in a large-scale network environment and investigating the effect of a dynamic whitelist generation approach on constructing a whitelist. In addition, future work could also include further verifying the IP verification module and combining contextual information with the list-based packet filter on packet filtration.

Acknowledgments. We thank the HoneybirdHK for providing the Honeypot dataset and supporting our work, and all the anonymous reviewers for their helpful comments.

References

1. Paxson, V.: Bro: A System for Detecting Network Intruders in Real-Time. Computer Networks 31(23-24), 2435–2463 (1999)
2. Scarfone, K., Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS). NIST Special Publication 800-94 (2007)
3. Roesch, M.: Snort: Lightweight Intrusion Detection for Networks. In: Proceedings of Usenix Lisa Conference, pp. 229–238 (1999)
4. Dreger, H., Feldmann, A., Paxson, V., Sommer, R.: Operational Experiences with High-Volume Network Intrusion Detection. In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 2–11 (2004)
5. Vigna, G., Kemmerer, R.A.: NetSTAT: A Network-based Intrusion Detection Approach. In: Proceedings of Annual Computer Security Applications Conference (ACSAC), pp. 25–34 (1998)
6. Valdes, A., Anderson, D.: Statistical Methods for Computer Usage Anomaly Detection Using NIDES. Technical Report, SRI International (January 1995)
7. Ghosh, A.K., Wanken, J., Charron, F.: Detecting Anomalous and Unknown Intrusions Against Programs. In: Proceedings of Annual Computer Security Applications Conference (ACSAC), pp. 259–267 (1998)

8. Snort, The Open Source Network Intrusion Detection System, http://www.snort.org/ (accessed on April 21, 2012)
9. Rivest, R.L.: On the worst-case behavior of string-searching algorithms. SIAM Journal on Computing, 669–674 (1977)
10. Isacenkova, J., Balzarotti, D.: Measurement and Evaluation of A Real World Deployment of A Challenge-Response Spam Filter. In: Proceedings of ACM SIGCOMM Conference on Internet Measurement Conference (IMC), pp. 413–426 (2011)
11. Sommer, R., Paxson, V.: Outside the Closed World: On using Machine Learning for Network Intrusion Detection. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 305–316. IEEE, New York (2010)
12. Lofgren, P., Hopper, N.: FAUST: Efficient, TTP-free Abuse Prevention by Anonymous Whitelisting. In: Proceedings of Annual ACM Workshop on Privacy in the Electronic Society (WPES), pp. 125–130 (2011)
13. Wireshark, http://www.wireshark.org/ (accessed on April 25, 2012)
14. Erickson, D., Casado, M., Mckeown, N.: The Effectiveness of Whitelisting: a User-Study. In: Proceedings of Conference on Email and Anti-Spam, pp. 1–10 (2008)
15. Yoon, M.K.: Using Whitelisting to Mitigate DDoS Attacks on Critical Internet Sites. IEEE Communications Magazine 48(7), 110–115 (2010)
16. IP Source Guard, http://www.cisco.com/en/US/docs/switches/lan/catalyst6500/ios/12.2SX/configuration/guide/ipsrcgrd.html (accessed on May 12, 2012)
17. Synopsys Verification IP, http://www.synopsys.com/Tools/Verification/FunctionalVerification/VerificationIP/Pages/default.aspx (accessed on May 12, 2012)
18. Chen, E.Y., Itoh, M.: A Whitelist Approach to Protect SIP Servers from Flooding Attacks. In: Proceedings of IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR), pp. 1–6 (2010)
19. Colasoft Packet Builder, http://www.colasoft.com/packet_builder/ (accessed on April 12, 2012)
20. Kim, T.H., Choi, Y.S., Kim, J., Hong, S.J.: Annulling SYN Flooding Attacks with Whitelist. In: Proceedings of International Conference on Advanced Information Networking and Applications Workshops, pp. 371–376 (2008)
21. McHugh, J.: Testing Intrusion Detection Systems: a Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. ACM Transactions on Information System Security 3(4), 262–294 (2000)
22. Meng, Y., Kwok, L.F.: Adaptive Context-aware Packet Filter Scheme using Statistic-based Blacklist Generation in Network Intrusion Detection. In: Proceedings of International Conference on Information Assurance and Security (IAS), pp. 74–79 (2011)
23. Li, J., Sung, M., Xu, J., Li, L.: Large-Scale IP Traceback in High-Speed Internet: Practical Techniques and Information-Theoretic Foundation. IEEE/ACM Transactions on Networking 16(6), 1253–1266 (2008)
24. Goodrich, M.T.: Efficient Packet Marking for Large-Scale IP Traceback. In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 117–126 (2002)
25. Jin, C., Wang, H., Shin, K.G.: Hop-Count Filtering: an Effective Defense Against Spoofed DDoS Traffic. In: Proceedings of ACM Conference on Computer and Communications Security (CCS), pp. 30–41 (2003)
26. Yaar, A., Perrig, A., Song, D.: Pi: A Path Identification Mechanism to Defend against DDoS Attacks. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 93–107 (2003)

On the Automated Analysis of Safety in Usage Control: A New Decidability Result

Silvio Ranise[1] and Alessandro Armando[1,2]

[1] Security and Trust, FBK-ICT, Trento, Italy
[2] DIST, Università di Genova, Italy

Abstract. We study the decidability of the safety problem in the usage control (UCON) model. After defining a formal model, we identify sufficient conditions for the decidability of the safety problem for UCON systems whose attributes are allowed to range over infinite domains and updates in one process may affect the state of another. Our result is a significant generalization of those available in the literature.

1 Introduction

Access control aims at protecting data and resources against unauthorized disclosure and modifications while ensuring access to authorized users. The usage control (UCON) model [12] extends access control by supporting *mutable attributes* and *continuity of enforcement*. Subjects and objects are associated with attributes that may be updated by usage processes; examples of attributes are identity, role, group membership, and the number of times a resource has been accessed. Continuity of enforcement means that a permission can be dynamically granted and revoked depending on some conditions on the (mutable) attributes.

In this paper, we study one of the fundamental problems of the UCON model and, more in general, of any protection model: the *safety problem* (see, e.g., [10]) that consists of checking the existence of the leakage of access permissions as a side-effect of the execution of usage control policies. The solution of a safety problem instance reveals whether a UCON policy preserves desired security properties, encoded as invariants, across changes to the (mutable) attributes. Failure in preserving any one of the properties implies the presence of a security breach that designers have the opportunity to correct before deployment; thereby greatly reducing the cost of debugging. To make this scenario viable in practice, automated tools for solving instances of the safety problem must be available that are capable of reasoning about a huge number of interleaving executions of UCON policies induced by the concurrent updates to the (mutable) attributes. The first step in the direction of building such tools is to identify classes of UCON policies with decidable safety problem. This has been studied before [16,17] for very restricted UCON policies in which attributes may take only finitely many values and concurrency is not allowed. To the best of our knowledge, no other decidability result for the safety of UCON policies is available and security analysis "*still remains an open issue in usage control*" [9].

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 15–28, 2012.

We make two contributions towards the development of comprehensive formal models for usage control that are amenable to automated analysis.

The first contribution is a **formal model** (inspired to that in [15] and based on the standard notion of labelled transition system [8]) of a significant subset of UCON policies in which (a) attributes may range over finite or infinite domains and (b) several usage processes run concurrently so that updates of attributes in one process may affect the state of others (e.g., causing revocation of a permission). As the second contribution, we identify a class of UCON policies that can be symbolically specified by using first-order (decidable) theories (it is well-known how to represent transition systems with first-order formulae, see again [8]) and find sufficient conditions to guarantee the **decidability of** their **safety problem**. Because of features (a) and (b) of our model (see first contribution above), the scope of our decidability result is significantly wider than those available in the literature. A symbolic model-checking procedure (based on Satisfiability Modulo Theories (SMT) solving [4]) is used in the proof of decidability and paves the way to the development of automated analysis techniques of UCON policies by leveraging both model checking and SMT solving techniques. Such techniques are crucial to assist in the design of UCON policies by

Plan of the paper. Section 2 introduces our UCON model and define the safety problem. Section 3 identifies a subclass of UCON systems for which it is possible to show decidability of the safety problem. Section 4 concludes and discusses related work. Omitted proofs can be found in the extended version of the paper at `http://st.fbk.eu/SilvioRanise/#Papers`.

2 A Formal UCON Model

Given a configuration of a system, an access control mechanism decides whether a subject is allowed a certain right on an object. For instance, given a set of permissions at a cloud service, the access control system must decide whether a certain employee of an organization can view a given table T in a corporate database *CDB*.

Usage Control (UCON) extends access control by allowing for mutable attributes and continuity of control. Mutable attributes are associated to subjects, objects, or the system and are updated as side-effects of a UCON mechanism. Continuity of enforcement means that the UCON mechanism can revoke a permission granted to a subject s on an object o when a certain condition—depending on the attribute values of s or o—is no more satisfied. For instance, the attribute n_T of the table T in the corporate database *CDB* can count how many employees are viewing T at the same time while the attribute t^e_T records for how long an employee e is viewing T. If n_T is larger than a fixed number, the UCON mechanism can revoke the permission of viewing T to the employee e whose attribute t^e_T is highest. In this way, more recent requests can be served, thereby allowing for a more sensible exploitation of T.

A UCON mechanism comprises subjects, objects, permissions, attribute values (identifying the part of a system state that is relevant to access control),

and functionalities that grant and revoke permissions to subjects on objects depending on conditions, constraining subjects or objects attributes. Since attributes are mutable, a UCON mechanism must also identify those operations that change the system state and specify how they update the attribute values. To process an access request, a new instance of the UCON mechanism, called UCON process, is created. Thus, we need to consider the situation in which several UCON processes run concurrently and the updates to attributes are interleaved. In this scenario, it may happen that the change of an attribute by one UCON process may imply the revocation of a permission by another one since its access condition may become false as a result of the update of the former. As an example, recall the situation described above in which the permission to view the table T in the corporate database *CDB* is revoked from an employee because of a new request by another employee.

Our model of UCON—called $UCON_A$ as we will only consider authorizations and, for simplicity, ignore obligations and conditions—is composed of the following basic ingredients: a set O of *objects*, a set S of *subjects*, a set R of *rights* (or permissions), a set I of UCON process identifiers. As it is customary in the literature about access control (see, e.g., [12]), we assume that $S \subseteq O$. The set $P := S \times O \times R$ of *permissions* consists of triples (s, o, r) meaning that subject $s \in S$ has right $r \in R$ on object $o \in O$. Each subject or object in O is associated with a (finite) sequence $\underline{a}$ of *attributes*; e.g., identifier, role, or credit amount. In turn, each attribute a in $\underline{a}$ (written $a \in \underline{a}$) takes values over a *domain* D_a, i.e. a set of elements; we write $\underline{D_a}$ (or simply $\underline{D}$ when $\underline{a}$ is clear from the context) for the sequence of domains associated with the attributes in $\underline{a}$. An *object-attribute mapping* is a sequence $\underline{m_a}$ (or simply $\underline{m}$ when $\underline{a}$ is clear from the context) of functions from O to $D_a \cup \{\perp_a\}$ with $\perp_a \notin D_a$ denoting that the value of the attribute a is unspecified for an object in O.

2.1 Object Labelled Transition System

We formally characterize how mutable attributes are updated by the operations that change the part of the system state that is relevant to usage control. We do this by using the standard notion of *labelled transition system* (see, e.g., [8]), that is a tuple $(\mathcal{S}, \mathcal{L}, \rightarrow)$ where $\mathcal{S}$ is the set of states, $\mathcal{L}$ is a set of labels, and $\rightarrow \subseteq \mathcal{S} \times \mathcal{L} \times \mathcal{S}$ is a ternary relation; usually, $(s, l, s') \in \rightarrow$ is written as $s \xrightarrow{l} s'$.

The *object labelled transition system* is the tuple $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$ where $\mathcal{S}_O$ is a set of pairs of the form $(AO, \underline{m})$, where AO is a finite sub-set of O and $\underline{m}$ is an object-attribute mapping such that $m_a(o) = \perp_a$ for every $o \in O \setminus AO$ and each $a \in \underline{a}$. The objects in AO are *active* and those in $O \setminus AO$ are *inactive*. The condition on $\underline{m}$ requires that the values of the attributes of inactive objects are unspecified while that of any attribute of an active object may have a value distinct from $\perp_a$ (i.e. it is specified). An element of $\mathcal{S}_O$ is an *object state*.

In several papers about usage control (e.g., [12]), operations that modify an object state, called *primitive operations*, are not specified explicitly. Although our UCON_A model is parametric with respect to primitive operations, here we consider the same primitive operations in [16] for the sake of concreteness: $\mathcal{L}_O$

contains $\mathsf{createobj}(o)$, $\mathsf{destrobj}(o)$, $\mathsf{update}(o, a, v_a)$ for $o \in O$, $a \in \underline{a}$, and $v_a \in D_a$. Intuitively, the effect of the action $\mathsf{createobj}(o)$ is to make the object o active, that of $\mathsf{destrobj}(o)$ is to make o inactive, and that of $\mathsf{update}(o, a, v_a)$ is to set the value of the attribute a of object o to the value v_a. Mathematically, to define the effects of these operations, we write $f \oplus \{x \mapsto y\}$ for the function that returns the same results of f on every input, except on x for which it returns the value y. For $cmd \in \mathcal{L}_O$, we define $\xrightarrow{cmd}_O$ by the following three (inductive) rules:

$$\frac{o \in O \setminus AO \quad m'_a = m_a \oplus \{o \mapsto \bot_a\} \text{ for } a \in \underline{a}}{(AO, \underline{m}) \xrightarrow{\mathsf{createobj}(o)}_{aux} (AO \cup \{o\}, \underline{m}')} \qquad \frac{o \in AO}{(AO, \underline{m}) \xrightarrow{\mathsf{destrobj}(o)}_{aux} (AO \setminus \{o\}, \underline{m}')}$$

$$\frac{o \in AO \quad a \in \underline{a} \quad v_a \in D_a \quad m'_a = m_a \oplus \{o \mapsto v_a\} \text{ and } m'_b = m_b \text{ for } b \in \underline{a} \setminus \{a\}}{(AO, \underline{m}) \xrightarrow{\mathsf{update}(o,a,v_a)}_{aux} (AO, \underline{m}')}$$

The first rule (on the left) says that the object o can become active (by adding it to the set AO of active objects) provided that it has been inactive so far and the value of the attributes of o are unspecified. The second rule (on the right) says that the object o can become inactive (by deleting it from the set AO) provided that o is active (i.e. it belongs to AO). The last rule says that the value of the attribute a (first premise) of the active object o is set to v_a, the values of the other attributes of o are unchanged as well as those of all the attributes of the other active objects. By letting $\rightarrow_O$ to be the union of $\xrightarrow{\lambda}_O$ over $\lambda \in \mathcal{L}_o$, we conclude the definition of the instance $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$ of the object labelled transition system inspired to [16].

Below, we extend $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$, called the *underlying* object labelled transition system, with authorizations (Section 2.2) and concurrent executions of usage processes (Sections 2.3 and 2.4).

2.2 Authorizations and UCON$_A$ Policies

We start by introducing the notion of *attribute predicate* as a Boolean-valued function on $\underline{D}$. Let p be an attribute predicate and $\underline{d}$ a tuple of $\underline{D}$, if $p(\underline{d})$ is true, then we say that $\underline{d}$ *satisfies* p, in symbols, $\underline{d} \models p$. Given an attribute predicate p, an object-attribute mapping $\underline{m}$, and an object $o \in O$, we say that o *satisfies* p *under* $\underline{m}$ iff $\underline{m}(o) \models p$. An *authorization* A is a pair (P_s, P_o) of finite sets of attribute predicates and *the pair* $(s, o) \in S \times O$ *satisfies* A *under the object-attribute mapping* $\underline{m}$ iff $\underline{m}(s) \models p_s$ and $\underline{m}(o) \models p_o$, for each $p_s \in P_s$ and $p_o \in P_o$, in symbols $\underline{m}(s, o) \models A$. If $A = (\emptyset, \emptyset)$, then $\underline{m}(s, o) \models A$ for any object-attribute mapping $\underline{m}$, subject s, and object o.

Authorizations and primitive operations are then combined in the notion of *UCON$_A$ policy*, that is an expression of the form

$$\ell(s, o, r) : PreA \mid OnA \rightsquigarrow pres \mid ons \mid posts$$

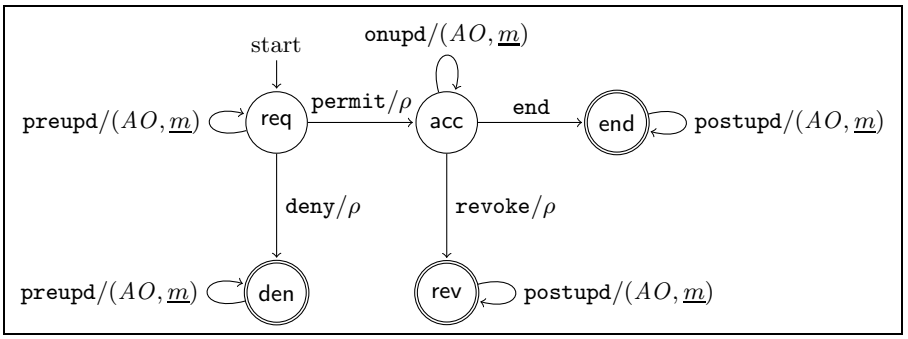


Fig. 1. The (extended) automaton of a usage process

where ℓ is a *policy identifier*, s is a subject, o an object, r a right, $PreA(s, o)$ and $OnA(s, o)$ authorizations, $pres, ons, posts$ (possibly empty and finite) sequences of primitive operations in $\mathcal{L}_O$. (To simplify notation, we assume that '|' binds stronger than '$\rightsquigarrow$.') The idea is that the UCON$_A$ policy ℓ is instantiated to establish if a subject s can get right r on object o. Because of continuity of enforcement, the request can be granted, denied, and revoked according to the authorizations *PreA* and *OnA* that are checked before and during access, respectively. Because of mutability, attributes (of active objects) are updated according to the primitive operations in the sequences *pres*, *ons*, and *posts* that are executed before, during, and after access, respectively.

2.3 A Semi-formal Account of Concurrent UCON$_A$ Processes

The informal characterization of the notion of UCON$_A$ policy given above can be made precise by viewing a UCON$_A$ mechanism as a finite collection of processes that—besides updating the state $(AO, \underline{m})$ of the underlying object labelled transition system $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$—maintain also the *permission mapping* $\rho : S \times O \rightarrow 2^R$, i.e. a function that associates pairs of (active) subjects and objects with sets of rights (notice that $\rho(s, o) = \emptyset$ if $s, o \notin AO$). More precisely, a usage process is an instance of a UCON$_A$ policy whose authorizations and primitive operations are executed according to the (extended) automaton in Figure 1 (adapted from [15]). From the discussion above, the transitions of the automaton can modify the state $(AO, \underline{m})$ of the underlying object labelled transition system $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$ or the permission mapping ρ. This is reflected in Figure 1 by the labels $\mathtt{l}/s$ of transitions: `l` is an action name and s is the "part" of the state that can be modified by `l`, i.e. either $(AO, \underline{m})$ or ρ. The execution of an instance of the automaton can be divided in three phases. In the first phase, before granting right r on object o to subject s (cf. transition `permit` from location req to location acc whose side-effect is to update ρ by adding the right r to the set of rights associated to the pair (s, o)), the authorization *PreA* is checked and the actions in *pres* are executed (cf. transition `preupd` labelling the self loop on location req whose side-effect is to modify the state of the underlying object

labelled transition system). In the second phase, access is granted, the authorization *OnA* is checked, and the operations in *ons* are executed (cf. transition `onupd` labelling the self loop on location `acc` whose side-effect is to modify the state of the underlying object labelled transition system): after the execution of each action, the authorization *OnA* is checked again. In the third phase (transition `end` from location `acc` to location `end` without side-effects), the actions in *posts* are executed and the usage process terminates (cf. transition `postupd` labelling the self loop on location `acc` whose side-effect is to modify the state of the underlying object labelled transition system). If at any time during the first phase, *PreA* no longer holds, access is denied (cf. transition `deny` from location `req` to location `den` whose side-effect is to update ρ by removing the right r from the set of rights associated to the pair (s, o) if the case). Similarly, if *OnA* no longer holds at any time during the second phase, access is revoked (cf. transition `revoke` from location `acc` to location `rev` whose side-effect is to update ρ by removing the right r from the set of rights associated to the pair (s, o)). Roughly, the locations $Q := \{\mathsf{req}, \mathsf{acc}, \mathsf{den}, \mathsf{rev}, \mathsf{end}\}$ of the automaton in Figure 1 identifies the following situations during usage processing: `req` and `end` mark the beginning and termination of an access request (respectively), `den` and `rev` correspond to denying access because authorizations do not hold before or during access (respectively), and `acc` is when a subject exploits the resources immediately after access has been granted.

2.4 A Formal Characterization of Concurrent UCON$_A$ Processes

To formalize the discussion above, we define a *UCON$_A$ (protection) mechanism* as a tuple $(I, O, S, R, \underline{a}, \underline{D}, L, UP)$ where I is a (countably infinite) set of identifiers, O is a (countably infinite) set of objects, $S \subseteq O$ is a set of subjects, R is a (finite) set of rights, $\underline{a} = a_1, ..., a_n$ is a finite sequence of attributes, $\underline{D} = D_{a_1}, ..., D_{a_n}$ is a finite sequence of domains, L is a finite set of policy identifiers (labels), and UP is a finite set of UCON$_A$ policies.

Example 1. To exemplify this notion, we present a scenario inspired by the enterprise information system described in [7,14]. In this system, directors and managers can award bonuses to other employees as well as hire or fire them. The policy of the company states that (R1) directors can give and take bonuses to managers or employees according to their performances, (R2) managers can do the same to employees that are neither managers nor directors, (R3) directors can promote or demote employees to or from managers, and (R4) managers may hire or fire employees.

Let $(I, O, S, R, \underline{a}, \underline{D}, L, UP)$ be a UCON$_A$ mechanism where $S = O$, $R = \{updb, hire, fire\}$, $\underline{a} = r, b, p$, $\underline{D} = \{dir, man, emp\}$, $\mathbb{R}, \{poor, avg, high\}$, $L = \{\ell_1, ..., \ell_{10}\}$, and UP contains the following policies (below, $\ell(s, o, r) : A \rightsquigarrow seq$ abbreviates $\ell(s, o, r) : A|(\emptyset, \emptyset) \rightsquigarrow seq|\epsilon|\epsilon$ and when A is a singleton, curly braces are omitted):

$$\ell_1(s, o, updb) : (r = dir, p = poor) \rightsquigarrow \mathsf{update}(b, o, 0)$$
$$\ell_2(s, o, updb) : (r = dir, p = avg) \rightsquigarrow \mathsf{update}(b, o, 20)$$
$$\ell_3(s, o, updb) : (r = dir, p = high) \rightsquigarrow \mathsf{update}(b, o, 80)$$

$\ell_4(s, o, updb) : (r = man, \{r = emp, p = poor\}) \rightsquigarrow \mathsf{update}(b, o, 0)$
$\ell_5(s, o, updb) : (r = man, \{r = emp, p = avg\}) \rightsquigarrow \mathsf{update}(b, o, 10)$
$\ell_6(s, o, updb) : (r = man, \{r = emp, p = high\}) \rightsquigarrow \mathsf{update}(b, o, 50)$
$\ell_7(s, o, chr) : (r = dir, r = man) \rightsquigarrow \mathsf{update}(r, o, emp)$
$\ell_8(s, o, chr) : (r = dir, r = emp) \rightsquigarrow \mathsf{update}(r, o, man)$
$\ell_9(s, o, hire) : (r = man, \emptyset) \rightsquigarrow \mathsf{createobj}(o); \mathsf{update}(r, o, emp)$
$\ell_{10}(s, o, fire) : (r = man, \{r = emp, p = poor\}) \rightsquigarrow \mathsf{destrobj}(o)$

for $s \in S$ and $o \in O$. The policies can assign three rights: *updb* for setting the value of the bonus, *hire* for hiring, or *fire* for firing an employee. The attribute r can take values *dir*ector, *man*ager, or *emp*loyee; the attribute b represents the amount of the bonus; and the attribute p represents the performance of the employee as *poor*, *avg* (average), or *high*. Policies ℓ_1, ℓ_2, ℓ_3 correspond to (R1), ℓ_4, ℓ_5, ℓ_6 to (R2), ℓ_7 and ℓ_8 to (R3), ℓ_9 and ℓ_{10} to (R4). □

Semantically, a UCON$_A$ mechanism $(I, O, S, R, \underline{a}, \underline{D}, L, UP)$ identifies a family of systems, each one composed of a finite collection of concurrently executing instances of the automaton in Figure 1, called *usage processes*, in which authorizations and sequences of primitive actions are those specified by the set UP of policies. Every finite sub-set of the set I identifies a finite collection of usage processes, derived from the policies in UP. Formally, given an underlying object labelled transition system $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$, the semantics of $(I, O, S, R, \underline{a}, \underline{D}, L, UP)$ is given by a family of labelled transition system $\upsilon = (\mathcal{S}_\upsilon, \mathcal{L}_\upsilon, \rightarrow_\upsilon)$ whose components are as follows.

The states of $\mathcal{S}_\upsilon$ are tuples of the form $(AI, (AO, \underline{m}), \rho, U)$ such that $(AO, \underline{m})$ is a state of the underlying object labelled transition system, ρ is a permission mapping, $AI \subseteq I$ is the finite set of *active* identifiers while those in $I \setminus AI$ are *inactive*, and U is a set of tuples, called *usage process states*, of the form

$$(q(\ell, id, s, o, r), PreA, OnA, pres, ons, posts)$$

where $PreA$ and OnA are authorizations, $pres, ons, posts$ are (finite and possibly empty) sequences of primitive operations, and $q(\ell, id, s, o, r)$ denotes the tuple (q, ℓ, id, s, o, r) where $q \in Q = \{\mathsf{req}, \mathsf{acc}, \mathsf{den}, \mathsf{rev}, \mathsf{end}\}$ (compare with the locations of the extended automaton of Figure 1), $\ell \in L$, $id \in I$, $s \in S$, $o \in O$, and $r \in R$. We also assume that $|U| = |AI|$,[1] i.e. there exists a bijection between active identifiers and usage process states. Intuitively, a state σ in $\mathcal{S}_\upsilon$ keeps track of the active objects and the values of their attributes, since σ contains the state of the underlying object labelled transition system $(\mathcal{S}_O, \mathcal{L}_O, \rightarrow_O)$. Furthermore, σ records the set AI of active identifiers and the progress made by the usage processes in elements of U, each one storing a location of the automaton in Figure 1 and the sequences of primitive operations that are still to be executed.

The second component of the labelled transition system υ is the set $\mathcal{L}_\upsilon$ of labels that contains the following elements: $\mathtt{preupd}(id)$, $\mathtt{deny}(id)$, $\mathtt{permit}(id)$, $\mathtt{idle}(id)$, $\mathtt{onupd}(id)$, $\mathtt{revoke}(id)$, $\mathtt{postupd}(id)$, $\mathtt{end}(id)$ for every $id \in I$.

[1] The operator $|\cdot|$ is overloaded and may denote the cardinality of a set or the length of a sequence. Similarly, $\emptyset$ denotes both the empty set and the empty sequence.

The elements in $\mathcal{L}_\upsilon$ (except for $\texttt{idle}(id)$ that is introduced in [6] and will be discussed below) correspond to the labels of transitions in the (extended) automaton of Figure 1. (The fact that labels are of the form $l(id)$ means that transition l is considered in the context of the instance (identified by) id of the extended automaton.) Intuitively, the labels in $\mathcal{L}_\upsilon \setminus \{\texttt{end}(id), \texttt{idle}(id) | id \in I\}$ identify *control transitions* performed by the system while $\texttt{end}(id)$ and $\texttt{idle}(id)$ label *security relevant transitions* performed by the user. The execution of $\texttt{end}(id)$ means that the user intends to terminate the usage process while $\texttt{idle}(id)$ models the situation in which a user remains idle despite the fact that he/she has gained a certain permission (imagine, e.g., the use of a web-site in which accessing resources and idle periods are interleaved). In the automaton of Figure 1 as well as in our formalization, the security relevant transition `tryaccess` considered in [15] is missing because we have chosen to model the permission request of a user as part of the initial state of the UCON_A system as follows. A state is *initial* when finitely many usage processes are in location req and are associated with some permission request (s, o, r) for $s \in S$, $o \in O$, and $r \in R$. Furthermore, the set of active objects is assumed to be finite, some of the attributes of the active objects are defined while those of the inactive objects are undefined. Finally, the permission mapping is the constant function returning the empty set of rights for every pair of subject and object.

The third and last component of the labelled transition system υ is the transition relation $\rightarrow_\upsilon$ which is the union of $\xrightarrow{\lambda}_\upsilon$ over $\lambda \in \mathcal{L}_\upsilon$. In the following, for lack of space, we list only two of the rules that (inductively) defines $\xrightarrow{\lambda}_\upsilon$ (all rules are in the extended version of this paper) and are the formal counterparts of the transitions in the automaton of Figure 1. For instance, the rule

$$\frac{\begin{array}{ll} u_{id} = (\mathsf{req}(\ell, id, s, o, r), \mathit{PreA}, \mathit{OnA}, \epsilon, \mathit{ons}, \mathit{posts}) & \underline{m}(s, o) \models \mathit{PreA} \\ u'_{id} = (\mathsf{acc}(\ell, id, s, o, r), \mathit{PreA}, \mathit{OnA}, \epsilon, \mathit{ons}, \mathit{posts}) & \rho' = \rho \oplus \{(s, o) \mapsto \rho(s, o) \cup \{r\}\} \end{array}}{(AI, (AO, \underline{m}), \rho, U \cup \{u_{id}\}) \xrightarrow{\texttt{permit}(id)}_\upsilon (AI, (AO, \underline{m}), \rho', U \cup \{u'_{id}\})}$$

corresponds to the transition `permit` from location req to acc (notice the side-effect on the permission mapping that adds the right since the authorization PreA is satisfied) and the rule

$$\frac{\begin{array}{ll} u_{id} = (\mathsf{acc}(\ell, id, s, o, r), \mathit{PreA}, \mathit{OnA}, \epsilon, on; \mathit{ons}, \mathit{posts}) & \underline{m}(s, o) \models \mathit{OnA} \\ u'_{id} = (\mathsf{acc}(\ell, id, s, o, r), \mathit{PreA}, \mathit{OnA}, \epsilon, \mathit{ons}, \mathit{posts}) & (AO, \underline{m}) \xrightarrow{on}_O (AO', \underline{m}') \end{array}}{(AI, (AO, \underline{m}), \rho, U) \xrightarrow{\texttt{onupd}(id)}_\upsilon (AI, (AO', \underline{m}'), \rho, U)}$$

corresponds to the self-loop labelled `onupd` in location `acc` (notice the side-effect on the state of the underlying object transition system resulting from the execution of the primitive action on in the sequence $on; \mathit{ons}$).

As anticipated above when discussing $\mathcal{L}_\upsilon$, we have added the action $\texttt{idle}(id)$ to model the situation in which a user remains idle despite the fact that he/she has gained a certain permission. This is formalized by the following rule

$$\frac{(\mathsf{acc}(\ell, id, s, o, r), \mathit{PreA}, \mathit{OnA}, \epsilon, \mathit{ons}, \mathit{posts}) \in U \quad \underline{m}(s, o) \models \mathit{OnA}}{(AI, (AO, \underline{m}), \rho, U) \xrightarrow{\texttt{idle}(id)}_\upsilon (AI, (AO, \underline{m}), \rho, U)}$$

corresponding to the situation in which the state of the UCON$_A$ mechanism is unchanged, provided that the authorization *OnA* holds. This rule and the one above whose conclusion is labelled by `onupd`(id) formalize continuity of access of UCON$_A$ mechanisms, for which ongoing authorization is active through the usage of the requested permission and the authorization *OnA* is repeatedly checked for sustaining access. In combination with mutability of attributes, the execution of a transition labelled `idle`(id) or `onupd`(id) can be interleaved with that of transitions of other usage processes that may change the value of objects attributes that are relevant for *OnA*. In other words, executions of transitions labelled by `idle`(id) or `onupd`(id) correspond to authorization checks performed periodically based on events. An interesting alternative (briefly discussed in [12]) would be to specify ongoing authorization checks based on time passing; this is left as future work.

2.5 The Safety Problem for UCON$_A$ Mechanisms

The safety problem amounts to establishing whether, from an initial state of a UCON$_A$ mechanism, a subject s can obtain the right r on an object o after the execution of a (finite) sequence of usage processes (i.e. after a sequence of primitive operations for updating attributes, or creating and destroying objects). To formally define this, we need to introduce the notion of *run* of a UCON$_A$ mechanism $\upsilon = (\mathcal{S}_\upsilon, \mathcal{L}_\upsilon, \rightarrow_\upsilon)$ as a finite sequence $\sigma_0, ..., \sigma_n$ of states such that $\sigma_k \rightarrow_\upsilon \sigma_{k+1}$ for each $k = 0, ..., n-1$. We write $\rightarrow_\upsilon^*$ for the reflexive and transitive closure of $\rightarrow_\upsilon$. The *safety problem* for UCON$_A$ mechanisms is defined as follows.

Instance: A UCON$_A$ mechanism $(I, O, S, R, \underline{a}, \underline{D}, L, UP)$, an initial state $\sigma = (AI, (AO, \underline{m}), \rho, U)$, and a finite set Π of permissions.
Question: Does there exist a labelled transition system υ induced by $(I, O, S, R, \underline{a}, \underline{D}, L, UP)$, a state $\sigma' = (AI', (AO', \underline{m'}), \rho', U')$ with $\sigma \rightarrow_\upsilon^* \sigma'$, and $r \in \rho'(s, o)$ for each permission $(s, o, r) \in \Pi$?

Example 2. In the context of the employee information system described in Example 1, an interesting security question is whether two managers can collude so that one can give a (positive) bonus to the other? The question can be recast as the following instance of the safety problem. Let $\sigma = (AI, (AO, \underline{m}), \rho, U)$ be an initial state where $m_r(o_1) = m_r(o_2) = man$ and for some ℓ_j $(j = 1, ..., 10)$ we have a tuple $(\mathsf{req}(\ell_j, id, s, o, r), PreA_{\ell_j}, seq_{\ell_j})$ in U and $\rho(s, o) = \emptyset$ for every $s, o \in O$, $r \in \{updb, h, fire\}$ with $PreA_{\ell_j}$ and seq_{ℓ_j} authorization and sequence of primitive operations, respectively, of the policies in UP (see Example 1). The problem boils down to determining whether there exists a sequence of transitions from σ to a state $\sigma' = (AI', (AO', \underline{m'}), \rho', U')$ such that $o_1, o_2 \in AO'$, $m'_b(o_1) > 0$, $m'_b(o_2) > 0$, and either $updb \in \rho'(o_1, o_2)$ with $m'_r(o_1) = man$ or $updb \in \rho'(o_2, o_1)$ with $m'_r(o_2) = man$. It turns out that this is possible by having two usage processes, one running an instance of policy ℓ_7 and one of policy ℓ_5 (or ℓ_6). With the instance of ℓ_7 some director o demotes a manager o_1 so that the manager o_2 who is subject of the instance of policy ℓ_5 (or ℓ_6) can give a positive bonus to o_1. □

As illustrated above, the capability of checking unsafety allows one to detect security breaches in UCON_A mechanisms. In the example, a collusion between two users permits to circumvent the security requirement (R2) stated in Example 1. Unfortunately, checking safety of UCON_A mechanisms is, in general, undecidable. In fact, it is possible to adapt the proof of undecidability in [16] to our framework (the notion of UCON system considered there is an instance of our notion of UCON_A mechanism). In the next section, we identify sufficient conditions to guarantee the decidability of the safety problem.

3 Decidability of Safety of Symbolic UCON_A Mechanisms

We want to identify sub-classes of UCON_A systems for which the safety problem is decidable by using the symbolic model checking approach to check safety properties of concurrent systems (see, e.g., [4]). For this, a preliminary step is to define a symbolic representation for UCON_A mechanisms. Given the long tradition of using first-order logic as the foundation of authorization languages (see, e.g., [10]) and the availability of efficient automated reasoners (e.g., SMT solvers) that offer a good starting point for developing automated analysis techniques (see, e.g., [4]), we use *logical (constraint) theories* as our symbolic representation. Intuitively, a logical theory [3] fixes the the meaning of some symbols (such as $+$ or $\leq$). The idea is to use logical theories to formally specify the algebraic structure of the values of the attributes. In this way, attributes range over the domains specified by logical theories and constraints (identifying sets of assignments to the variables occurring in the constraints that make them true) can be used to specify attribute predicates and, ultimately, authorizations. Similar observations also holds for the specification of primitive operations; a constraint expresses the relationship between the values of the mutable attributes immediately before and after the execution of a primitive operation.

According to the symbolic model checking approach, solving the safety problem reduces to the (exhaustive) exploration of the space of values assigned to attributes, called *states*, symbolically represented as logical constraints, that are induced by all possible executions of the primitive operations specified in UCON_A policies. Procedures performing such an exploration are called *reachability* procedures and work by repeatedly computing the set of reachable states obtained from the execution of operations from the initial (*forward* reachability) or the goal (*backward* reachability) set of states, that halt when the set of reachable states is detected to be a *fix-point*. The degree of automation and complexity of reachability procedures is highly dependent on the symbolic representation chosen to represent the systems under analysis. While identifying conditions for the decidability of fix-point checks is relatively easy, the crux is the termination of the reachability procedure.

Below, we develop our decidability result using logical theories as our symbolic representation in the framework of many-sorted first-order logic [3]. We introduce the notion of symbolic UCON_A mechanisms that uses logical theories to specify the domains over which attributes of UCON_A policies range and constraints for

authorization and primitive operations (Section 3.1). Then, we identify sufficient conditions on theories and constraints to guarantee the decidability of the safety problem for such a class of UCON$_A$ mechanisms (Section 3.2).

3.1 Symbolic UCON$_A$ Mechanisms

A *symbolic* UCON$_A$ mechanism is a tuple $(T_{Id}, T_{Obj}, T_{Right}, T_{Att}, \underline{att}, T_{Policy}, SUP)$ whose components are the following. T_{Obj} and T_{Id} are theories of equality (i.e. the only interpreted symbol is $=$ with the usual meaning of an equivalence relation) over the sorts *Obj* and *Id*, respectively; T_{Obj} contains also the unary predicate *isSubj* whose interpretation is a subset of that of the sort *Obj* (the interpretations of *Obj* is the set O of objects and that of *isSubj* is the sub-set S of O). T_{Right} and T_{Policy} are theories of an enumerated data-type over the sorts *Right* and *Policy*, respectively, whose interpretations are the sets R of rights and L of policy identifiers, respectively. The theory T_{Att} of the attributes is obtained as the union of the theories T_a's over attributes; the signature of each T_a (for a an attribute) is assumed to contain just one sort symbol Dom_a. Intuitively, each theory T_a specifies the algebraic structure of the values in the interpretation D_a of Dom_a that the attribute a may assume (e.g., T_a can be Linear Arithmetic or a theory of an enumerated data-type). $\underline{att}$ is a finite sequence of functions such that $att_a : Obj \to Dom_a$ for each attribute a. A symbolic UCON$_A$ policy in the (finite) set *SUP* is an expression of the form $\ell(s, o, r) : PreA \mid OnA \rightsquigarrow pres \mid ons \mid posts$, where ℓ is a constant (as defined in first-order logic, see, e.g., [3]) of sort *Policy*, s, o are constants of sort *Obj*, r is a constant of sort *Right*; *PreA* and *OnA* are constraints in the logical theory T_{Att}; and *pres*, *ons*, *posts* are finite sequence of operations. We illustrate this notion with an example.

Example 3. The symbolic UCON$_A$ system $(T_{Id}, T_{Obj}, T_{Right}, T_{Att}, \underline{att}, T_{Policy}, SUP)$ that formalizes the employee information system described in Example 1 is the following: T_{Id} and T_{Obj} are theories of equality, T_{Right} is the theory of the enumerated data-type $\{Right, \{updb, hire, fire\}$, $T_{Att} := T_r \cup T_b \cup T_p$ with T_r the theory of the enumerated data-type $\{dir, man, emp\}$), T_b the theory of the usual less-than-or-equal relation $\leq_b$ over the reals $\mathbb{R}$, T_p the theory of the enumerated data-type $\{poor, avg, high\}$, and T_{Policy} the theory of the enumerated data-type $\{\ell_1, ..., \ell_{10}\}$. The sequence $\underline{att}$ contains the functions att_r, att_b, and att_p returning the value of the attributes r, b, and p for a given object. Finally, the set *SUP* contains $\ell(s, o, r) : A^* \rightsquigarrow seq$ for each policy $\ell(s, o, r) : A \rightsquigarrow seq$ in *UP* defined in Example 1, where A^* is obtained from $A = (A_s, A_o)$ as follows: replace each attribute a occurring in A_s and A_o with variable x_a and y_a, respectively, and then conjoin all the resulting expressions. □

3.2 Decidability of a Class of Symbolic UCON$_A$ Mechanisms

We now present the conditions on attribute theories and constraints that guarantee the decidability of the safety problem.

Condition (C1) concerns the primitive operation createobj and is twofold. On the one hand, we assume that the subjects of usage processes are active. This restriction is also assumed in [16] and implies that the authorization *PreA* may depend only on the values of the subject attributes. On the other hand, we require that createobj can only be applied to the object o of the usage process obtained as an instance of the policy $\ell(s, o, r) : \cdots$. This seems to be more restrictive than the 'bounded creation' assumption in [16] (i.e. the sequence "a subject can create an object which in turn can create another object" is of finite length). However, we observe that the enterprise information system of Example 1 as well as the UCON systems considered in [16] (e.g., Administrative Role Based Access Control and Digital Right Management systems) can be expressed in this restricted notion of symbolic UCON$_A$ system. This suggests that the decidability result stated below covers a significantly large portion of practically relevant systems. *Condition* (C2) is about the logical theories: attributes may take values over finite or infinite domains with "simple" algebraic structure. Formally, the notion of "simple" is characterized by the class of *universal and relational* theories, i.e. theories whose axioms are sentences containing only universal quantifiers and constant or predicate symbols, but no functions. For example, the reals with the usual ordering is a simple algebraic structure in this sense (this theory is used for instance in Example 3 above). However, as soon as we add the usual arithmetic operations of addition and subtraction, the structure is no longer simple.

Theorem 1. *The safety problem for a symbolic UCON$_A$ mechanism* $(T_{Id}, T_{Obj}, T_{Right}, T_{Att}, \underline{att}_a, T_{Policy}, SUP)$ *is decidable if conditions (*C1*) and (*C2*) hold.*

The proof, that can be found in the extended version of this paper, consists of showing that a backward reachability procedure (using constraints to represent goals and UCON$_A$ policies) always terminates for the identified class of symbolic UCON$_A$ mechanisms. Interestingly, the proof is structured along the lines of standard proofs of termination for backward reachability procedures in the symbolic model checking approach (see, e.g., [4]).

We now discuss the complexity of the UCON$_A$ safety problem and the backward reachability procedure used in the proof of Theorem 1. When solving an instance of UCON$_A$ safety problem, only a fixed number of instances of the automaton in Figure 1 needs to be considered (this is shown in the proof of Theorem 1). The number of primitive operations in the instances of the automaton in Figure 1 can be bounded by a given number (this can be easily computed by inspection of the UCON$_A$ policies). Thus, it is possible to determine the set η of all possible executions of a UCON$_A$ mechanism. (Notice that the set of states that can be reached by the executions in η may still be infinite if one of the attributes range over an infinite domain.) Now, one can reduce the executability of a certain execution in η to solving a satisfiability modulo the attributes theory problem (this is a consequence of the proof of Theorem 1), that can be shown to be decidable under the assumptions of Theorem 1. Solving an instance of the UCON$_A$ safety problem can thus be reduced to checking the satisfiability modulo the attributes theory of the formula α obtained by conjoining that representing the initial state, that representing the final state, and that obtained

by taking the disjunction of the formulae encoding the possible executions in η. Now, checking the satisfiability of α modulo the attributes theory is NP-hard as it subsumes the propositional satisfiability problem and it is well-known to be NP-complete. Thus, also the UCON_A problem is NP-hard. The situation may look even worse when considering the complexity of the backward reachability procedure used in the proof of Theorem 1. In fact, such a procedure needs to solve a satisfiability problem after each pre-image computation to check for fix-point, i.e. it solves a NP-hard problem per iteration. Despite this, we believe it is possible to obtain a good behaviour of backward reachability on realistic UCON_A policies(such as those in [13]) by reusing our experience in developing the efficient analysis technique for ARBAC policies in [1]. We leave this to future work; here, we only notice that it is not possible to reduce the UCON_A safety problem to the safety problem of ARBAC policies for two reasons. First, it is impossible to encode attributes ranging over infinite domains with the simple (i.e. non-parametric) roles of [1], that may take only finitely many values. Second, it is difficult (if possible at all) to simulate the concurrent executions of a finite collection of instances of the automaton in Figure 1 by means of the administrative operations in [1].

4 Discussion

Several papers have proposed formal models for UCON with various limitations. Lamport's Temporal Logic of Actions is used in [15] to provide a complete specification of the behaviour of a single usage process. In [6], Interval Temporal Logic is used to model the interactions of several concurrent usage processes. In [5], a simplified UCON model is developed in the context of a timed concurrent constraint programming paradigm. An approach based on a process algebra is used in [11] to formally describe a comprehensive model for UCON systems. All these works do not discuss the use of automated techniques for analysing policy specifications in the proposed model. In [13], an extension of Linear Time Temporal logic is used to formalize usage control and a state-of-the-art model checker is exploited for analysis, although no decidability result is derived.

To the best of our knowledge, few works—that are most closely related to ours—have studied the decidability of the safety problem for UCON policies. In [16,17], only two simplified formalizations of the UCON model are given, in which attributes may take finitely many values and concurrency is greatly restricted (usage processes cannot influence the executions of others). The decidability of the safety problem for such restricted classes of policies is derived. Our decidability result (Theorem 1) generalizes those in [16,17] in three respects: (1) attribute values may range over finite or infinite domains, (2) we allow for concurrent execution of primitive operations, and (3) the results in [16,17] hold for situations in which the access decision is taken before or during the exercise of the requested right; our model covers both situations even in combination.

Generic model checking techniques for analysing concurrent systems (see, e.g., [8] for an overview) can be applied to solving the safety problem of UCON_A mechanisms. However, no decidability result available for classes of concurrent

systems covers that in Theorem 1. Even the decidability results in [4] cannot be applied because UCON_A mechanisms are parametric in two dimensions (process identifiers and objects) while the systems in [4] are mono-dimensional.

As already remarked above, the main limitation of UCON_A mechanisms is the lack of conditions—that are environment restrictions to be valid before or during access—and obligations—that are actions to be performed before or during access. Indeed, it would be interesting to include them in our model and possibly derive new decidability results. On the one hand, conditions can be easily incorporated by adding a distinguished object for the environment and leaving updates of the related attributes unspecified (possibly adding invariants [4] to constrain their values for obtaining more precise analysis results). On the other hand, incorporating obligations is more complex and left to future work.

References

1. Alberti, F., Armando, A., Ranise, S.: ASASP: Automated Symbolic Analysis of Security Policies. In: Bjørner, N., Sofronie-Stokkermans, V. (eds.) CADE 2011. LNCS, vol. 6803, pp. 26–33. Springer, Heidelberg (2011)
2. De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Samarati, P.: Access control policies and languages. IJCSE 3(2), 94–102 (2007)
3. Enderton, H.B.: A Mathematical Introduction to Logic. Academic Press, New York (1972)
4. Ghilardi, S., Ranise, S.: Backward reachability of array-based systems by smt solving: Termination and invariant synthesis. LMCS 6(4) (2010)
5. Jagadeesan, R., Marrero, W., Pitcher, C., Saraswat, V.: Timed constraint programming: A declarative approach to usage control. In: PPDP. ACM (2005)
6. Janicke, H., Cau, A., Zedan, H.: A note on the formalization of UCON. In: SACMAT. ACM (2007)
7. Kleiner, E., Newcomb, T.: On the Decidability of the Safety Problem for Access Control Policies. In: AVoCS. ENTCS, pp. 91–103 (2006)
8. Kröger, F., Merz, S.: Temporal Logic and State Systems. Springer (2008)
9. Lazouski, A., Martinelli, F., Mori, P.: Usage control in computer security: A survey. Computer Science Review 4, 81–99 (2010)
10. Li, N., Tripunitara, M.V.: Security analysis in role-based access control. ACM Trans. Inf. Syst. Security 9(4), 391–420 (2006)
11. Massonet, P., Arenas, A., Martinelli, F., Mori, P., Crispo, B.: Usage control for trust and security in next generation grids. In: At Your Service. MIT Press (2008)
12. Park, J., Sandhu, R.: Towards usage control models: Beyond traditional access control. In: SACMAT, pp. 57–64. ACM (2002)
13. Pretschner, A., Rüesch, J., Schaefer, C., Walter, T.: Formal Analyses of Usage Control Policies. In: Int. Conf. on Av., Rel. and Sec., pp. 98–105 (2009)
14. Zhang, N., Ryan, M.D., Guelev, D.P.: Evaluating Access Control Policies Through Model Checking. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 446–460. Springer, Heidelberg (2005)
15. Zhang, X., Parisi-Presicce, F., Sandhu, R., Park, J.: Formal Model and Policy Specification of Usage Control. ACM TISSec 8(4), 351–387 (2005)
16. Zhang, X., Sandhu, R., Parisi-Presicce, F.: Safety Analysis of Usage Control Authorization Models. In: ASIACCS. ACM (2006)
17. Zhigang, Z., Jiandong, W., Yuguang, M.: Study and Safety Analysis of UCON_{onA} Model. In: 1st Int. Ws. on Database Technology and App., pp. 103–106 (2009)

Attestation of Mobile Baseband Stacks

Steffen Wagner, Sascha Wessel, and Frederic Stumpf

Fraunhofer Research Institution AISEC, Garching, Germany
{steffen.wagner,sascha.wessel,frederic.stumpf}@aisec.fraunhofer.de

Abstract. Distributed denial of service (DDoS) attacks from a large number of compromised mobile devices are a major threat to mobile networks. In this paper, we present a concept, an architecture, and a protocol for a hardware-based attestation which enables mobile devices to efficiently prove that their baseband stack is still trustworthy. Our attestation mechanism enables verification of the baseband stack without using expensive asymmetric cryptographic operations, maintains the ability to update (or recover) the baseband binary, and allows the network to enforce a certain version, state, or configuration of the baseband at network connect. Our approach represents an efficient method to block devices with a compromised baseband stack and thus prevents distributed denial of service attacks to mobile networks.

Keywords: Attestation, Baseband Stack, Trusted Computing, Distributed Denial of Service, Mobile Network.

1 Introduction

Mobile cellular networks provide the infrastructure for location-independent global communications, i.e., phone calls, short messages, and data connections. Many people, businesses, and governments heavily rely on those means to communicate with each other, their customers, and counterparts in different parts of the world. That is why mobile communication networks are considered part of the *critical infrastructure* and need to be secure and highly available, because malfunction or failure can lead to potentially high damage and costs.

In contrast to mobile network nodes which are fully controlled by their operators, the billions of mobile devices are usually beyond their sphere of influence, thus untrusted. For that reason, network operators generally issue a smart card, which securely stores the pre-shared authentication information to access the network and thereby establishes mutual trust. However, as complexity and functionality of software components increase, it becomes easier for an adversary to compromise and remotely control a mobile device [8]. In particular, the baseband stack is an interesting attack target, because it implements the software stack to communicate with the mobile network. If an attacker is able to exploit baseband stack vulnerabilities of a large number of mobile devices, critical attacks on the network, such as distributed denial of service (DDoS) attacks, become possible.

One approach to protect a network from attacks by compromised devices is based on *Trusted Computing* and a Mobile Trusted Module (MTM) [14]. An

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 29–43, 2012.

MTM, a mobile version of a Trusted Platform Module (TPM) [15], can be used to securely handle cryptographic keys which can be marked as non-migratable to prevent extraction and external usage. The MTM also provides mechanisms to realize *attestation* which enables a trusted platform to prove that no adversary has tampered with its software. However, these security services provided by the MTM are primarily used by applications that are executed on the application processor. The MTM cannot be directly used to prove the trustworthiness of the baseband stack and to prevent attacks on the network infrastructure, which are caused by manipulated baseband stacks running on the baseband processor.

In this paper, we present a hardware-based attestation—more specifically, a concept, an architecture, and a protocol—for mobile baseband stacks. Our attestation enables a mobile device to efficiently prove its trustworthiness towards the network without the need for expensive asymmetric cryptography. Instead, symmetric operations transfer the result of the attestation from the prover to the verifier. Based on the attestation, the mobile network can grant (or restrict) the access to critical core components. As a result, the risk and the potential damage of attacks on the network from devices with a compromised baseband stack can be limited. It even enables the network to enforce a certain baseband version, which prevents attacks that exploit vulnerabilities in a (prior version of the) baseband stack in order to attack the network.

The remainder of this paper is structured as follows. In Sect. 2, we present related work and Sect. 3 briefly describes a generic hardware and software architecture of a mobile phone. Sect. 4 explains the infrastructure of mobile networks and possible attack scenarios. In Sect. 5, we describe the notation, cryptographic keys, and the concept of our attestation protocol. Finally, we provide a detailed security analysis in Sect. 6 and briefly conclude in Sect. 7.

2 Related Work

Most existing attestation protocols [9,16] for mobile devices, such as smart phones, enable these devices to prove their integrity to a remote verifier. The main idea of these protocols is based on existing Trusted Computing concepts, such as *authenticated boot* and *remote attestation*, as specified by the Trusted Computing Group (TCG). However, existing protocols [4,7,11] primarily focus on the software executed on the application processor, such as the operating system and the applications, but not the baseband stack running on the baseband processor. Thus, they cannot be directly used to prove the trustworthiness of the baseband stack. As a consequence, they are not able to prevent attacks on the network infrastructure in case an attacker was able to compromise the baseband stack of a larger number of mobile devices. However, research shows that such kind of attacks are a realistic scenario, since there already exist a number of demonstrated exploits for vulnerabilities in mobile baseband stacks [8,13].

In addition, the previously proposed attestation protocols [9,16] mainly rely on expensive asymmetric cryptographic operations, such as signing. However, even

with a dedicated MTM or TPM as cryptographic coprocessor, such operations are, in general, inefficient and may not be suitable for mobile devices.

3 Mobile Device Architecture

Mobile devices in a 3G or 4G network are a very heterogeneous mass of devices. However, individual devices always consist of two main (hardware) components: the *baseband (processor)* and the *application (processor)*. Fig. 1 shows such a common multi-CPU architecture for smart phones and 3G USB modems, where the processors communicate via a serial line or shared memory. In this architecture, the baseband CPU is usually the master and the application CPU is the slave. So, the baseband software stack may have full access to sensitive data of the smart phone operating system, but not the other way around. Thus, for common architectures, it is not necessary to measure the integrity of the application to guarantee the integrity of the baseband—at least in theory.

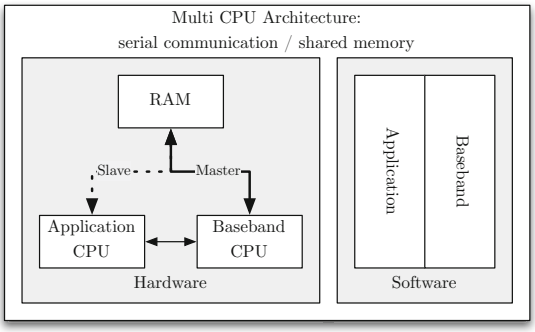


Fig. 1. Common mobile device architecture

3.1 Baseband Hardware Architecture

The baseband hardware in mobile devices usually consists of the following parts: radio frequency (RF) front end, analog baseband, and digital baseband consisting of a digital signal processor (DSP) and an ARM system on chip (SoC). In this paper, only the ARM SoC (baseband processor) is considered.

To the baseband processor, an exchangeable Universal Subscriber Identity Module USIM is connected, which is a smart card issued by the provider. The Universal Subscriber Identity Module (USIM) holds in its read-only memory (ROM), an operating system, and the security algorithms for authentication and key generation. In its EEPROM, it stores specific identity information, namely the International Mobile Subscriber Identity (IMSI) and Temporary Mobile Subscriber Identity (TMSI) as well as a secret K_{i}, which is shared with the provider.

However, today's mobile devices completely lack a comparable secure element to identify the device itself. The existing device-unique

International Mobile Equipment Identity (IMEI), for instance, is not stored securely, hence needs to be considered untrusted. For our concept, we therefore propose to extend the hardware architecture with an MTM. In contrast to existing solutions, we propose to directly connect the MTM to the baseband processor.

3.2 Baseband Software Components

Today's baseband software is usually a small real-time operating system which is responsible for parts of layer 1 (hardware specific physical layer) and everything above. For layer 2 and 3, it provides an hardware independent software stack with nested implementations of all 2G/3G/4G layers.

For the sake of simplicity, we divide the baseband in the following two parts: the *baseband binary* (B) and *baseband information* (BI) including device- or operator-specific configuration data. Although there might exist other binaries, such as fail-safe or backup binaries, only one baseband stack is running on the baseband processor. A boot loader, which is usually very small, loads the baseband binary from memory. We assume that the loader is stored in ROM to securely boot the baseband stack and authenticate updates that need to be cryptographically signed by the *baseband vendor* (BV).

For our protocol, we extend the boot loader and the baseband stack with the functionality to communicate with the MTM. Additionally, the USIM software now executes critical parts of the minimal TCG Software Stack (TSS) internally.

4 Mobile Network Infrastructure and Attack Analysis

In this section, we first briefly describe the architecture of a mobile network and show that mobile devices with a compromised baseband stack can inflict potentially high damage to the network.

4.1 Network Architecture

In general, a cellular mobile network is composed of a Core Network (CN), several Radio Access Networks (RANs), and User Equipment (UE) [1]. The UE wirelessly connects to a RAN, which is interconnected with the CN, to access the user's Home Network (HN), usually via a Service Network (SN).

The UE comprises the Mobile Equipment (ME), typically a mobile device with the necessary radio and protocol functionality, and the USIM, which securely stores the authentication information, mainly a shared cryptographic key, to access the mobile network. In order to access the network, the USIM and the network run an Authentication and Key Agreement (AKA) [3,10], which is a standardized challenge-response protocol that uses symmetric cryptography.

The RAN usually consists of transceiver stations called Node B (3G), which are managed by a Radio Network Controller (RNC), or Evolved Node B (4G), respectively, which embed their own controller functionality. These stations are connected to management components and support gateways, especially to a

Mobile Switching Center (MSC) with the Visitor Location Register (VLR) in a 3G and a Mobility Management Entity (MME) in a 4G network.

Apart from the interface and administration components for RANs, the CN mainly provides the Home Location Register (HLR) or Home Subscriber Server (HSS) depending on the network as well as the Authentication Center (AuC), which manage the relevant information for user authentication and accounting. In particular, the security credentials to access the network, e.g., the pre-shared key K_{i} between the USIM and the network, are securely stored in the AuC.

4.2 Attacks on Mobile Networks

Mobile networks are an interesting attack target. They belong to the critical infrastructure, because they provide world-wide communication services though a network of global access nodes. Furthermore, they heavily rely on centralized core components like the HSS and AuC for authentication, as described in the previous section, which could lead to complete network failure in the case of a successful attack against those components.

Based on the centralized architecture of mobile communication networks, an adversary might, for instance, try to attack the network components directly. However, a direct attack against the network is not very promising, since the core components usually only communicate with other trusted network components. Nevertheless, an attacker could still be able to attack the critical components indirectly by launching a DDoS attack.

For such an attack, the adversary needs to successfully compromise a large number of mobile devices, e.g., by sending non-specification-compliant SMS messages, which exploit certain weaknesses of the message parser [8]. The attacker could also try to manipulate or replace the baseband stack. If the attacker, for instance, can convince a mobile device to download a malicious version of the baseband binary and install it, the device is compromised as well.

With thousands of mobile devices under their control, attackers could launch a DDoS attack. In contrast to local attacks via low-layer access channels, e.g., RACHell [6], such global attacks can potentially produce a system-wide critical overload in the backend components of the network [13]. For example, if all the compromised devices drop off the network [3, 5.3.8 Detach procedure] and simultaneously re-connect again [3, 5.3.2 Attach procedure], the HSS might not be able to handle all the authentication requests [8]. Another example is the call-forwarding functionality, which is also handled by the HSS. If an attacker can change the settings for a large number of mobile devices at the same time, the overload could crash the HSS and the network could fail completely.

As a result, we have to acknowledge that attacks on mobile networks based on manipulated baseband stacks are realistic and they can inflict high damage, potentially a complete network failure.

5 Providing Verifiable Proof for the Trustworthiness of Mobile Baseband Stacks

To prevent attacks on networks based on compromised or non-specification-compliant baseband stacks, we propose an attestation protocol to verify the trustworthiness of a mobile device before it can communicate with the core components of a mobile network, such as the HSS.

The main idea is that only trustworthy mobile devices are allowed to fully access the critical components of a network. To demonstrate its trustworthiness, the baseband stack running on a mobile phone has to prove its authenticity and integrity. If the attestation procedure fails, the network only allows limited access, e.g., to download a trustworthy version of the baseband stack, which is signed by its vendor, to replace the compromised one. That way, the proposed attestation protocol allows the mobile device to recover from malicious modifications and protects the network from attacks by compromised mobile devices.

In the following sections, we first define the notation (Sect. 5.1) and the cryptographic keys (Sect. 5.2). We then explain the concept in Sect. 5.3 and present a description of our protocol in Sections 5.4 and 5.5.

5.1 Notation

A cryptographic *hash function* H is a one-way function with collision and preimage resistance that compresses input data of virtually arbitrary length to a fixed sized output of length l, that is $H : \{0,1\}^* \rightarrow \{0,1\}^l$. Applying the hash function H to data m is denoted as $H(m)$, which generates a hash h.

A *message authentication code MAC* is a function that calculates a *message digest* with fixed length l for an input m with virtually arbitrary size based on a secret key K: $MAC(K, m) = dig$. The resulting digest provides information that can be used to verify both the integrity as well as the authenticity of the message. An *HMAC* is a specific way to construct a MAC based on a hash function, i.e., $HMAC(K,\ m) = H((K \oplus opad) \,||\, H((K \oplus ipad) \,||\, m))$, where $||$ denotes concatenation, $\oplus$ the exclusive or, *opad* the outer and *ipad* the inner pad.

A particular system state is represented by a set of *integrity measurements* stored in the (currently up to 24) hardware-protected platform configuration registers (PCRs) of an MTM. Such a set of PCRs describing a system state is referred to as *platform configuration* $P := (PCR[i = 0], \ldots, PCR[i = p])$. After a system reset, the contents of all PCRs is set to zero, i.e., $PCR[i] \leftarrow 0 \quad \forall i < 24$. To store a fresh integrity measurement μ in a PCR with index i, the current value is combined with the fresh measurement value using $PCRExtend(PCR[i], \mu)$, which is specified as $PCR[i] \leftarrow SHA1(PCR[i] \,||\, \mu)$ [15].

Wrapping a cryptographic key K with a public asymmetric key pk to a specific platform configuration P is denoted as $\{K\}^P_{pk}$ and essentially encrypts the key. To decrypt the wrapped key with the corresponding private key sk, the actual configuration P' needs to match exactly the specified configuration P.

To encrypt and bind arbitrary data m to a platform configuration P, the MTM essentially provides a `TPM_Seal` command, which is referred to as *Seal* for the sake of simplicity. With *Unseal*, which is short for `TPM_Unseal`, the MTM/TPM can decrypt the data m if the system is in a state which matches the specified platform configuration P. Given a non-migratable asymmetric key $K_{\text{seal}} = (pks, sks)$, we denote the result of *sealing* data m to the platform configuration P with $\{m\}^{P}_{pks} = Seal(P, pks, m)$. To *unseal* the sealed data $\{m\}^{P}_{pks}$, it is required that the actual platform configuration P' is equal to the specified platform configuration P: $m = Unseal(P' = P, sks, \{m\}^{P}_{pks})$.

5.2 Cryptographic Keys

We define a non-migratable asymmetric wrapping key $K_{\text{wrap}} = (pk, sk)$ and sealing key $K_{\text{seal}} = (pks, sks)$, where pk and pks are the public keys while sk and sks are the secret keys of the respective keys. Both asymmetric keys are securely generated and stored inside the MTM.

With the public key pks, a public signing key $pkSig_{BV}$ is sealed to a platform configuration P_L, where all PCRs are selected, but have the value zero, except for $PCR[0]$, which stores the measurement value of the boot loader L. As a result, the sealed key $\{pkSig_{BV}\}^{P_L}_{pks}$ can only be unsealed by the boot loader, which is stored in ROM and executed first. This unsealed key $pkSig_{BV}$ is used in case of an update to verify a signature sig of the baseband update before installation.

With the pubic key pk, an asymmetric integrity key K_{INT} is wrapped to a trusted platform configuration P_B as $\{K_{\text{INT}}\}^{P_B}_{pk}$, where B denotes the baseband. This wrapped key is used in our protocol to verify the integrity of the baseband. It is important to note that the platform configuration P_B invalidates P_L, which means the platform configuration P_L can no longer be used as a valid platform configuration for cryptographic operations, e.g., to unseal a key.

Together with the authentication data A_{seal} for the sealing key K_{seal}, both keys are stored in flash memory in encrypted form (sealed or wrapped) during initialization. However, having A_{seal} in flash is not a security problem, because the key K_{seal} is protected by P_L, thus it is only available to the boot loader L.

The USIM holds the key K_{i} and an attestation key K_{ATT}, which are both shared with the AuC, as well as the passphrase A_{wrap} for wrapping key K_{wrap}.

5.3 Concept and Main Ideas

In contrast to existing hardware-based attestation protocols, which are often based on asymmetric cryptographic operations provided by an MTM [16], we propose a symmetric approach to efficiently prove the trustworthiness of a mobile device, in particular, its baseband stack.

As most existing protocols, we rely on an authenticated boot process starting from a Core Root of Trust for Measurement (CRTM): The current software binary calculates a hash of the following binary in the boot chain and stores it as an integrity measurement value in a platform configuration register of the TPM or MTM before executing the measured binary.

In our concept, the boot loader acts as CRTM for the sake of simplicity, which is why it must be stored in ROM as indicated in Fig. 2 (top left), which shows our system architecture. Together with the baseband binary (top right), Fig. 2 also depicts a boot procedure in the top and the relevant hardware components, namely the flash memory (bottom left), the MTM (bottom center), the USIM (bottom right), and the baseband processor with RAM, in the bottom half.

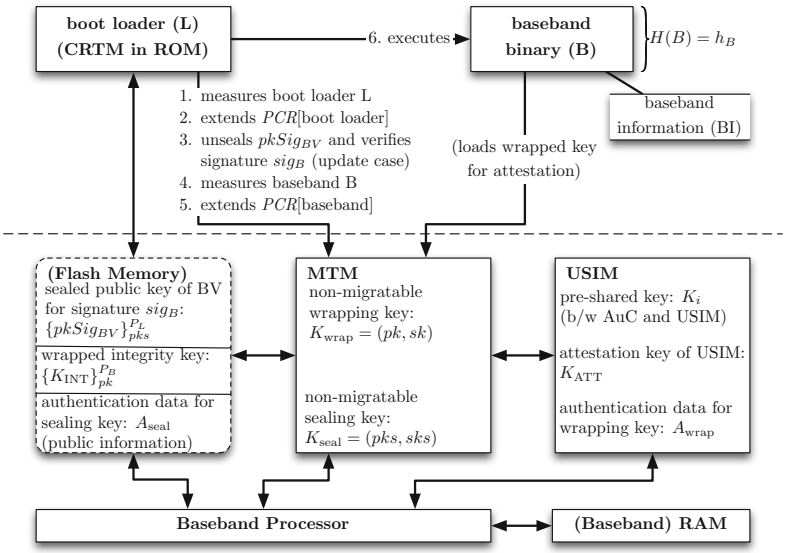


Fig. 2. System Architecture (based on Fig. 1)

In the boot procedure, the loader L first measures itself and extends the PCR[boot loader] (steps 1 and 2) creating the platform configuration P_L. In case of an update, the loader unseals the public signing key $pkSig_{BV}$, verifies signature sig of the new baseband binary with the unsealed $pkSig_{BV}$ (step 3) and re-wraps the integrity key K_{INT}. In the process, the unseal operation implicitly validates the integrity of the boot loader L (represented by P_L) and the public signing key verifies the new baseband with its signature. After that, the boot loader measures the baseband (step 4) and extends the PCR[baseband] (step 5), which creates the trusted platform configuration P_B and invalidates P_L. Finally, it executes the baseband binary (step 6).

For a remote attestation, the TPM or MTM normally signs the list of PCRs representing the current platform configuration and sends it to a remote verifier. Based on the values (and a so called *measurement log*), the remote party can then decide whether the platform is still trustworthy. We call this mechanism *explicit attestation*, because the complete measurement log needs to be transferred.

For our protocol, however, we adapt the concepts of *implicit attestation*, which does not need to transfer the measurement log. Instead, implicit attestation usually relies on some pre-shared (authentication) information, such as a sealed symmetric key or hash chain [7], which can only be accessed, if the platform is still trustworthy. This approach is, for instance, used to validate whether the boot loader is in a trusted state before a new baseband binary is loaded in case of an update (step 3). So, as long as the prover can successfully authenticate itself, the verifier has implicit proof of the integrity of the prover's system.

However, most existing implicit attestation protocols still rely on relatively expensive asymmetric cryptographic operations, such as signing or unsealing a sealed key. That is why we propose a more efficient approach based on symmetric cryptographic operations to implicitly prove the trustworthiness of a mobile device, especially its baseband stack.

The main idea is that the USIM only grants access to the attestation key K_{ATT} which is necessary to calculate an attestation response for the network if the baseband stack is trustworthy. To prove its trustworthiness, the baseband merely needs to load the integrity key K_{INT}, which is cryptographically bound (wrapped) to a trusted platform configuration P_B based on a signed baseband stack. Note that the initial wrap operation is necessary only once during initialization and thus has no direct influence on the overall efficiency of our attestation protocol. More important, the MTM only needs to actually unwrap the wrapped key if the key is not yet decrypted and loaded, because the key itself is never used for security critical operations. Finally, to be able to securely verify the baseband stack, we moved the calculation of authentication value, which is an HMAC based on A_{wrap} and needed to load the key K_{INT}, inside the USIM. So, the baseband has to request the correct authentication value from the USIM before it can load the key inside the MTM. If the load operation was successful, the USIM can verify the HMAC-authenticated result, which is protected by A_{wrap}.

5.4 Integrity Verification of the Baseband Stack

For a local attestation (integrity verification) between the baseband stack (prover) and the USIM (verifier), which is depicted in detail in Fig. 3, the baseband system (center) loads the wrapped key $\{K_{\mathrm{INT}}\}_{pk}^{P_B}$ into the MTM (right) and the USIM (left) verifies the HMAC-authenticated result. However, since we moved the HMAC calculation for the authentication and verification (steps 3 and 10) inside the USIM, all security critical operations are performed inside one of the hardware secure elements.

Usually, a dedicated trusted software stack is responsible to calculate and assemble the necessary parts of the command, such as the authentication value *parentAuth* (pA) for the wrapping key K_{wrap} (Fig. 3, step 3), and send the complete command structure to the MTM. In our protocol, however, the main idea is to move this calculation of the authentication value (pA, steps 2 and 3), which authorizes the use of the wrapping key K_{wrap}, inside the USIM. That way, the passphrase A_{wrap} never leaves the secure environment of the USIM firmware. As a result, the USIM needs to securely generate the required authentication value pA

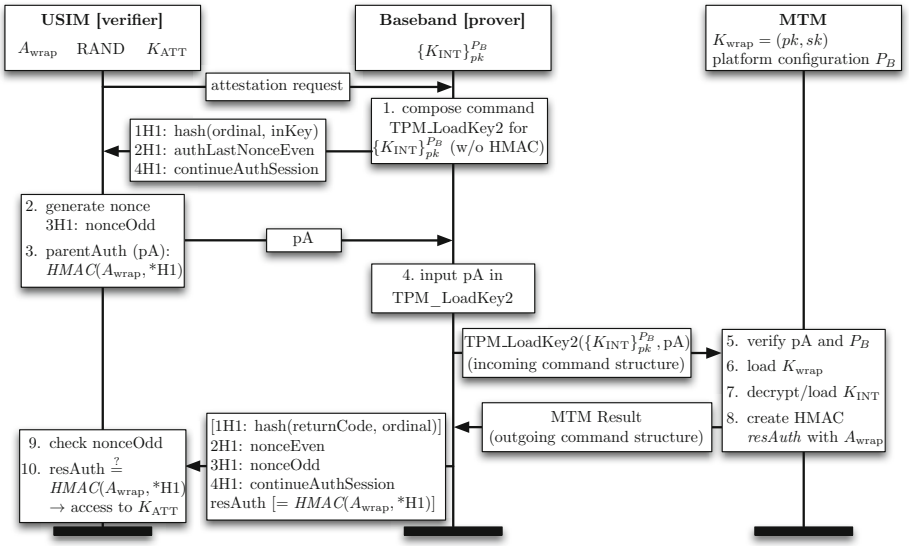


Fig. 3. Attestation of Baseband towards USIM

on behalf of the usual software component, which can be implemented in software and easily integrated into the existing USIM firmware. The value for *parentAuth* is generated from concatenated HMAC inputs (denoted by *1H1* to *4H1*) as

$$pA = HMAC(A_{wrap},\ 1H1\ ||\ 2H1\ ||\ 3H1\ ||\ 4H1)\,, \tag{1}$$

$$\begin{aligned} \text{where}\quad & 1\text{H1} = H(\text{TPM\_ORD\_LoadKey2}\,||\,\{K_{\text{INT}}\}_{pk}^{P_B}) \\ & 2\text{H1} = authLastNonceEven \\ & 3\text{H1} = nonceOdd, \text{ and} \\ & 4\text{H1} = continueAuthSession, \end{aligned}$$

according to the TCG specification [15, p. 72].

When the MTM receives the load command, it internally verifies the authentication value pA and matches the specified platform configuration P_B against the actual platform configuration P' (step 5). If the equation $P_B = P'$ holds, the MTM loads the key. For efficiency reasons, the MTM should only verify the pre-conditions, e.g., the platform configuration and authentication data, and not actually decrypt the key if the key is already loaded. The MTM then calculates a result message, which includes a specified return code, e.g., `TPM_SUCCESS`, the *nonceOdd*, and a second HMAC *resAuth* to authenticate the response (step 8).

To complete the attestation procedure, the USIM receives the result of the load operation $TPM\_LoadKey2(\{K_{\text{INT}}\}_{pk}^{P_B}, pA)$ and merely needs to verify the return message: For that purpose, the USIM compares the output *nonceOdd*

with the input *nonceOdd*, which must be exactly the same and prevents replay attacks (step 9). By recalculating and checking the HMAC *resAuth* (and the return code), the USIM can efficiently verify whether the key was correctly loaded, thus, stating that P_B matches P' (step 10). The fresh HMAC *resAuth'* is calculated again according to Equation 1, where

$$\begin{aligned} 1\text{H1} &= H(\mathit{returnCode} \,||\, \text{TPM\_ORD\_LoadKey2}) \\ 2\text{H1} &= \mathit{nonceEven} \\ 3\text{H1} &= \mathit{nonceOdd}, \text{ and} \\ 4\text{H1} &= \mathit{continueAuthSession} \end{aligned}$$

as specified by the TCG [15, p. 72]. If the load operation has been successful, which is indicated by the *returnCode*, the verifier has implicitly proven that the baseband stack is still unmodified and has not been compromised. As shown in Fig. 3, the USIM now allows access to the attestation key K_{ATT}, which is limited to the current AKA protocol run indicated by the random number *RAND*.

5.5 Generation of Authentication Vectors

Based on the result of a local baseband attestation, the USIM (now prover) is able to provide proof of the baseband's trustworthiness towards the network (verifier). We only need to slightly modify the authentication vectors (AVs) used in the AKA protocol. Depending on the network type (3G or 4G), the AVs are usually generated as

$$\mathit{UMTS\ AV} := (\mathit{RAND} \,||\, \mathit{XRES} \,||\, \mathit{CK} \,||\, \mathit{IK} \,||\, \mathit{AUTN}) \text{ or} \tag{2}$$

$$\mathit{EPS\ AV} := (\mathit{RAND} \,||\, \mathit{XRES} \,||\, K_{\text{ASME}} \,||\, \mathit{AUTN}), \tag{3}$$

where *RAND* is a random number, *XRES* is the pre-calculated (expected) authentication result, *CK* is a confidentiality and *IK* an integrity key, and *AUTN* an authentication token. These components are calculated as depicted in Fig. 4, where f1 and f2 are MAC and f3 to f5 as well as KDF are key derivation functions.

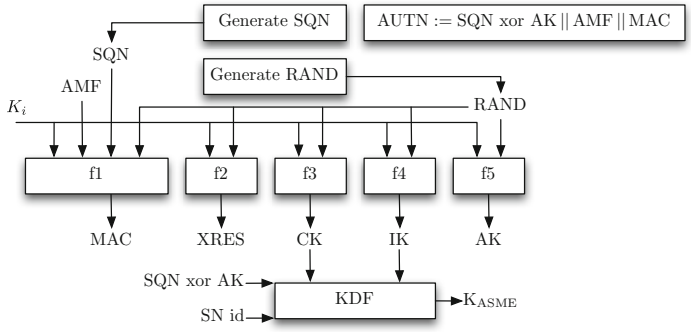


Fig. 4. Generation of Authentication Vectors (adapted from [2,3,5])

In our concept, we add an expected attestation value (*XATT*) to the AV, which allows the Service Network to verify the trustworthiness of the mobile device. This additional attestation value is generated from the random number *RAND* with a dedicated attestation key K_{ATT} (which is only available, if the local baseband attestation has been successful), that is

$$XATT = HMAC(K_{ATT}, RAND). \tag{4}$$

Since the symmetric attestation key K_{ATT} is only known to the HSS (and the USIM, of course), the Home Network has to pre-calculate the HMAC for the Service Network. That way, the SN can compare *ATT* (from the USIM) with the expected attestation value *XATT* without the knowledge of K_{ATT}.

We also send a second attestation value *ATTB* from the mobile device to the Home Network, which is generated based on the hash value of the baseband stack h_B and some information *BI*, for instance, about the version, state, and configuration of the baseband. These values can be used by the home network to further evaluate the baseband stack and enforce a certain version or configuration.

As shown in Fig. 5, we define the following attestation-based access policy: If the response $RES = \mathrm{f2}_{K_i}(RAND)$ from the USIM matches the expected response *XRES* and the attestation value $ATT = HMAC(K_{ATT}, RAND)$ also corresponds with the expected value *XATT*, the mobile device can fully access the network (steps 8–9). However, if the (local) attestation fails, the network only grants limited access, e.g., to download a signed recovery version to replace the modified baseband stack. By sending the hash of the baseband h_B and the

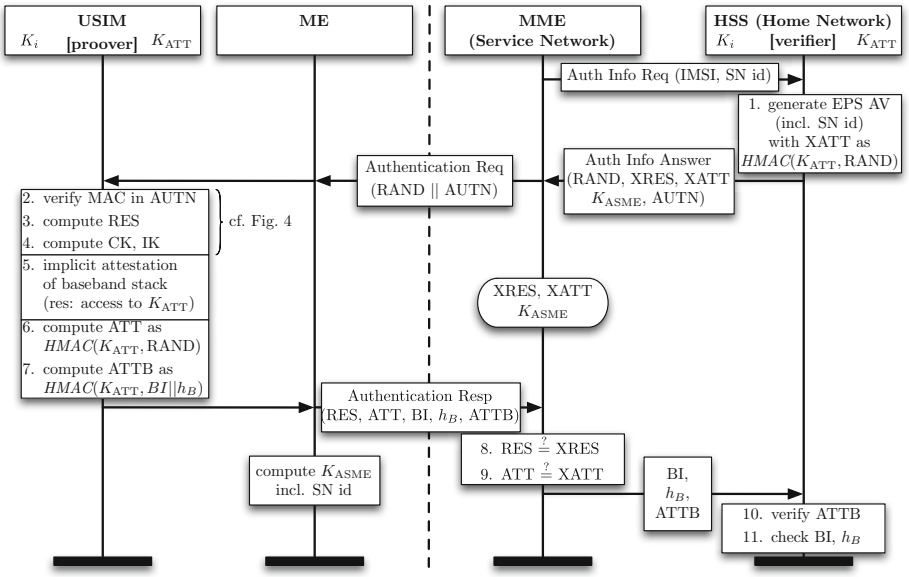


Fig. 5. AKA-based Attestation of Baseband/USIM towards the Network (simplified)

baseband information *BI*, which are protected by the second attestation value *ATTB* (step 7), the Home Network can evaluate the configuration of the baseband in detail (steps 10–11). As a consequence, particular services or operations involving critical network components could be allowed (or denied). The Home Network could even enforce a certain baseband version by simply evaluating the baseband version in *BI* and restricting access for unsupported versions.

6 Security Analysis

In this section, we now analyze the security of our proposed attestation protocol. We mainly consider software attacks, whereas hardware attacks, such as TPM cold boot attacks, are by nature and definition out of scope. As most existing protocols [16], we start from the premise (for the sake of simplicity) that the platform configuration reflects the actual state of the baseband stack at any time. We also assume that it is not possible to forge a trusted platform configuration by exploiting bugs, such as buffer overflows, although that requires either a periodical or an on-demand measurement architecture, such as IBM's IMA [12].

In our first scenario, the attacker attempts to extract the cryptographic keys. However, that is not possible, because the symmetric keys K_{i}, K_{ATT}, and the authentication data A_{wrap} are securely stored inside the USIM. The asymmetric keys K_{wrap} and K_{seal} are non-migratable, thus never leave the MTM.

The attacker could also try to replace the sealed or wrapped keys, namely $\{pkSig_{BV}\}_{pks}^{P_L}$ and $\{K_{\mathrm{INT}}\}_{pk}^{P_B}$. In the first case, the sealed public signing key can only be unsealed while the boot loader is executed and the P_L is not yet invalidated by P_B. Since the boot loader is stored in ROM, it is always executed first and cannot be modified. As a result, the adversary cannot seal a different public key, which would successfully verify a signature for a manipulated baseband update. In the second case, the attacker might try to wrap an integrity key to an insecure platform configuration, e.g., with no PCRs selected, to manipulate the baseband stack without the attestation protocol noticing. However, this is not possible, because the authentication data A_{wrap} is stored inside the USIM.

In the next scenario, the adversary actually manipulates the baseband binary to attack the network. However, since the baseband binary is measured by the boot loader before it is executed, the manipulation is reflected in the platform configuration P'_B. As a result, the MTM cannot load the wrapped key, the attestation fails (because of the return code), and the USIM denies access to attestation key K_{ATT}. That mean the attestation value *ATT* cannot be calculated correctly and the network only allows fail-safe access to network, which can effectively prevent the attack. In the case, where the attacker manipulates the baseband binary, but replays an old MTM result message in order to make the USIM believe that loading the integrity key was successful, the USIM simply needs to check the *nonceOdd* (Fig. 3, page 38, step 9). Since this value is random and only known to the USIM (Fig. 3, step 2), the attestation fails, because the replayed MTM result message has a different *nonceOdd*.

In our last scenario, the adversary might try to forge the attestation value *ATT* in order to access and attack the network with a compromised baseband

stack. However, if the attacker is able to capture the random number *RAND*, the authentication token *AUTN*, and the authentication response *RES*, it is still not possible to calculate the correct attestation value. The attacker has no knowledge about the attestation key K_{ATT}, which is securely stored in the USIM. Even in the case, where the adversary combines the authentication response *RES* with an old attestation value *ATT*', the network only grants limited access. Since the attestation value *ATT* is an HMAC over the current random number *RAND*, the pre-calculated attestation value *XATT* does not match *ATT*' (Fig. 5, page 40, step 9), so the attestation fails. The network only allows fail-safe access and an attack on the critical network components, such as the HSS, is prevented.

7 Conclusion

With today's increasing use of mobile communication, which might even rise in the near future, attacks from a larger number of mobile access devices with a compromised baseband stack are a serious threat to mobile networks. To limit the risk of potentially high damage, we presented a hardware-based implicit attestation protocol, which enables mobile devices with an MTM to prove the trustworthiness of their baseband stack towards the mobile network. Furthermore, the network is able provide different access levels based on the result of the attestation. It can even enforce a certain baseband version. Our security analysis shows that this way the network can limit exposure to compromised baseband stacks and reduce the risk of attacks from manipulated devices.

Acknowledgments. Parts of this work were supported by the German Federal Ministry of Education and Research (BMBF) under grant number 01BY1011.

References

1. 3rd Generation Partnership Project (3GPP): TS 23.002, Network Architecture. Technical Specification (1999-2012)
2. 3rd Generation Partnership Project (3GPP): TS 33.102, 3G security; Security architecture. Technical Specification (1999-2012)
3. 3rd Generation Partnership Project (3GPP): TS 33.401, System Architecture Evolution (SAE); Security architecture. Technical Specification (1999-2012)
4. Chen, L., Landfermann, R., Löhr, H., Rohe, M., Sadeghi, A.R., Stüble, C.: A Protocol for Property-Based Attestation. In: STC, pp. 7–16. ACM (2006)
5. Forsberg, D., Horn, G., Moeller, W.D., Niemi, V.: LTE Security. Wiley (2010)
6. grugq: Base Jumping: Attacking the GSM baseband and base station
7. Krauß, C., Stumpf, F., Eckert, C.: Detecting Node Compromise in Hybrid Wireless Sensor Networks Using Attestation Techniques. In: Stajano, F., Meadows, C., Capkun, S., Moore, T. (eds.) ESAS 2007. LNCS, vol. 4572, pp. 203–217. Springer, Heidelberg (2007)
8. Mulliner, C., Golde, N., Seifert, J.P.: SMS of death: from analyzing to attacking mobile phones on a large scale. In: Proceedings of the 20th USENIX Conference on Security, SEC 2011, p. 24. USENIX Association, Berkeley (2011)

9. Nauman, M., Khan, S., Zhang, X., Seifert, J.-P.: Beyond Kernel-Level Integrity Measurement: Enabling Remote Attestation for the Android Platform. In: Acquisti, A., Smith, S.W., Sadeghi, A.-R. (eds.) TRUST 2010. LNCS, vol. 6101, pp. 1–15. Springer, Heidelberg (2010)
10. Niemi, A., Arkko, J., Torvinen, V.: Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA). RFC 3310
11. Sadeghi, A.R., Stüble, C.: Property-based attestation for computing platforms: caring about properties, not mechanisms. In: NSPW, pp. 67–77. ACM (2004)
12. Sailer, R., Zhang, X., Jaeger, T., van Doorn, L.: Design and implementation of a TCG-based integrity measurement architecture. In: USENIX Security Symposium, SSYM 2004. USENIX Association, Berkeley (2004)
13. Traynor, P., et al.: On cellular botnets: measuring the impact of malicious devices on a cellular network core. In: Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS 2009. ACM, New York (2009)
14. Trusted Computing Group (TCG): Mobile Trusted Module (MTM) Specification
15. Trusted Computing Group (TCG): Trusted Platform Module (TPM) Specification
16. Wen, Y., Peng, X., Chen, S., Zhao, H.: A Secure Access Approach of UMTS Terminal Based on Trusted Computing. In: Proceedings of the Second International Conference on Networks Security, Wireless Communications and Trusted Computing, NSWCTC 2010, USA, 5 pages (2010)

A Scalable Link Model for Local Optical Wireless Networks

Tae-Gyu Lee and Gi-Soo Chung

Korea Institute of Industrial Technology (KITECH), Ansan, Korea
tigerlee88@empal.com, gschung@kitech.re.kr

Abstract. Current wireless protocols employ monotonous link model although optical wireless channel is being changed rapidly and essentially as is the infrared channel. The plain link model cannot be regarded as the physical characteristic of the optical wireless transmission links. Thus, it only evokes inefficient channel use and insufficient link transfer performance. This paper proposes an advanced link model for improving the performance of data transfer in optical wireless networks. Data frame propagation on link is considered for two link issues, i.e., resource efficiency and link access hit ratio issue. The link resource allocation scheme must be carefully designed so that the resource can be efficiently utilized in order to improve link transmission performance.

There are two major contributions; first, this paper presents a dynamic link model which is optimized for local optical wireless network. Various essential characteristics of optical wireless channel are channel speed, bandwidth, error ratios, etc. Second, the optical wireless channel provides a new link allocation method that considers multiple constraints in an advanced link model, intended to minimize link delivery time while developing opportunistic nature of wireless channel for good throughput link with less packet loss. Finally, this paper successfully achieves an effective link error-recovery method through the link allocation algorithm.

Keywords: Link model, optical wireless, transfer protocol, fault tolerant, local network.

1 Introduction

Lately, as mobile computing devices such as smart phones, smart devices, etc. become popular, the utilization of wireless network resource is becoming an important issue as well. However, the radio channel resource is physically limited and gradual depletion of available wireless resources is taking place. Therefore, efficient utilization techniques of wireless network resources are required. For this purpose, many resource allocation and management techniques are being developed for radio channel utilization.

The development of technology of optical wireless transport network using optical materials like LED, LD, etc is recognized as a major challenge. In particular, IrDA with infrared radio channels which is compatible with the existing wireless-LAN has

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 44–57, 2012.

been used for a general local area network. However, because data transmission link model for the IrDA optical wireless channel techniques relies on limited features of a typical wireless channel, it has limited availability for the efficient use and performance improvement of optical wireless channels [1][2][3][4].

An existing wireless network as Wi-Fi, Bluetooth, etc. provokes constant bandwidth, average transmission delay, and a relatively uniformed error rate [1][2]. Therefore, the traditional link transmission model is an average formal link one after an analysis of the physical characteristics of the wireless channel and link parameters such as the channel bandwidth, transmission time unit, and transmission error rate was averagely determined. However, the network parameters of transmission bandwidth, transmission delay, and error rate are dynamically changing depending on the rapid change of optical wireless environments because the physical characteristics of optical wireless channel are also changed by mobile environment of optical wireless terminals. Therefore, the existing average formal links model is not suitable for an optical wireless transport model [5][6][7].

This study first analyzes the data transfer properties of the optical wireless resources based on the existing link model. At the same time, it suggests a new link model optimized for the physical signal characteristics of optical wireless resources, and an optical wireless channel allocation scheme based on a new link model.

The main contributions of this study are twofold as follows. First, this paper presents a dynamic link model, which is optimized for local optical wireless networks. Various essential characteristics of optical wireless channel are considered for channel speed, bandwidth, error ratios, etc. Second, it provides a new link allocation scheme taking into account multiple constraints in an advanced link model, attempting to minimize link delivery time while developing opportunistic nature of wireless channel for good throughput link with less packet loss.

This paper will be described by the following sequences. Chapter 2 describes characteristics of communication link and models of short-range radio channel. Chapter 3 proposes the optical wireless link model and link allocation scheme based on optimization of optical wireless channels. Chapter 4 evaluates the proposed model compared with existing models and analyzes the performance of the proposed link allocation scheme. Finally, Chapter 5 concludes this work and future researches.

2 Link Model on WLAN

2.1 Physical Wireless Channel Characteristics

What is the best way to link two devices without a cable? This simple idea has now blossomed into several industries offering a bewildering assortment of products and protocols. In this section, I will explain the world of wireless communications mechanisms. The three most popular standards for short-range wireless data communication are IrDA, Bluetooth, and Wi-Fi. Each allows battery-powered devices to communicate without wires. Wi-Fi and Bluetooth with typical wireless channel characteristics have the standardized (uniformed) transmission characteristics such as constant periodic signals, constant bandwidth, and constant failure rates.

In 1994, Ericsson Mobile Communications began research on a radio module that could link mobile phones and accessories, especially headsets. Bluetooth-enabled wireless headsets started to emerge in 2000, but component cost, power usage, and even regulatory barriers prevented widespread adoption. Since then, cost and power usage has gradually shrunk, making Bluetooth a valuable add-on feature for high-end PDAs and mobile phones [12][13].

The first IEEE 802.11 specification was introduced in 1997 with the primary goal of providing wireless LAN access. At first, component costs arose where expensive, interlope was chancy, and security was a major concern. Together, these factors prevented widespread adoption. But, over time, component cost has dropped, many security concerns have been addressed, and new specification versions (such as 802.11b, 802.11a, and 802.11g) have emerged that increase throughput. In 1999, the Wi-Fi Alliance was launched to certify implementations and alleviate interoperability concerns. Because of the large physical range and "always-on" connection model, Wi-Fi technology consumes a lot of power, limiting its use in PDAs, phones, and other lightweight mobile devices [8][9].

The IrDA was launched in 1993 as a cable replacement technology. However, as the industry developed, the IrDA realized that it was necessary to provide specifications that went beyond the basics of cable replacement. Today, virtually every PDA and smart phone shipped supports IrDA, as do many mobile phones, laptops, printers, and other products [1][2].

We can separate aspects of wireless performance into several categories: connection delay, which is the time required to discover and establish connectivity; communication latency, which is the time taken to deliver the data through the network; and finally the effective throughput, which is the actual transfer speed (often much less than the native data rate allowed by the media). All the parameters were chosen generally conform to the DFIR physical definition in the IEEE802.11 standard [9].

Radio and infrared are complementary transmission media, and different applications favor the use of one or the other medium. Radio is favored in applications where user mobility must be maximized or transmission through walls or over long ranges is required and may be favored when transmitter power consumption must be minimized. Infrared is favored for short-range applications in which per-link bit rate and aggregate system capacity must be maximized, cost must be minimized, international compatibility is required, or receiver signal-processing complexity must be minimized [10][11].

Bluetooth chops its 1 Mbit/s data rate into tiny 625 microsecond slices. This hurts its effective throughput, but gives it two important advantages. First, it is resistant to interference at particular frequencies, since any given packet can be retransmitted quickly at a different frequency. Second, its low latency makes reliable, telephone-quality audio connections possible [12].

In IrDA, latency is variable, depending on how frequently the primary device polls the secondary. In many implementations, a period of polling with no data exchange causes the primary to back off its polling frequency to as long as 500ms. This can sometimes cause a noticeable delay, but it also lowers power usage dramatically. The wavelength band between about 780 and 950 nm is presently the best choice for most

applications of infrared wireless links, due to the availability of low-cost LED's and laser diodes (LD's), and because it coincides with the peak responsibility of inexpensive, low-capacitance silicon photodiodes [10].

Conventional wireless transfer link model is based on the standard IEEE802.11 model. All of Wi-Fi, Bluetooth and IrDA consist of the layered structures of physical and link layers in wireless local networks. The wireless network devices employ an access control protocol of TDMA to share a wireless communication link.

2.2 Typical Wireless Link Model

The transfer model in typical local wireless networks consists of the first physical layer and the second link layer. The link layer supports logical transfer path which transfers bit frames, and the physical layer transfer signal consisting of physical wireless transfer path.

The link layer provides monotonous data transmission channel, which periodically provides fixed time slot to any host. The network runs under a time-division multiple access (TDMA) slotted framework, and we assume all nodes are perfectly synchronized. The time frame consists of fixed-size time slots for data, and fixed-size time slots for control messages. During the period of one time frame, we assume block fading channel that remains relatively constant. Scheduling decisions are taken by all nodes in the network simultaneously at the beginning of each time frame at the control phase, and stay unchanged until the next frame.

However, physical layer provides dynamic transfer signal channel due to irregular signal power and quality characteristics of wireless physical channel. The PHY layer employs the adaptive modulation and coding techniques (AMC), where there are a finite number of transmission modes, each of which corresponds to a unique modulation and coding scheme and one particular interval of the received signal to interference plus noise ratio (SINR). Furthermore, in order to reduce the interference to adjacent concurrent transmissions, increase the frequency reuse and channel capacity, the nodes are equipped with directional antennas.

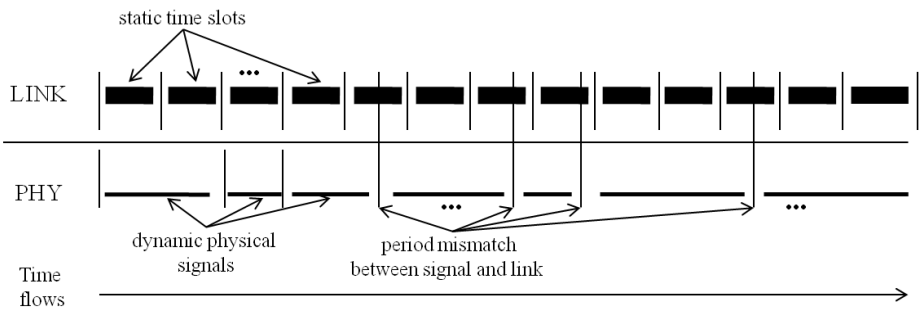


Fig. 1. Typical wireless link model

Fig. 1 shows the hierarchical structure of typical TDMA wireless link model. The structure makes efficiency of transfer channel worsen because of severe mismatches of slot period between dynamic physical channel and static link channel. Pervious link model make degrade the success ratio of host link access by the frequent mismatches and rapidly increase link access delay. Depending on static slot time, it also increases the inefficiency of link resource by increasing frame loss within a slot period.

This work proposes a new model and channel allocation scheme which harmonizes the channel period characteristics between dynamic physical channel and static link channel to improve the host access ability and channel efficiency problems.

2.3 Wireless Link Parameters

This paper considers link access delay, link slot time, link throughput (or bandwidth), and frame error ratio as performance metric parameters. It also assumes that all hosts share the single link channel for transferring data frames. Consider a wireless centralized network, which comprises a set of wireless mobile station or mobile terminal, denoted as MS and a set of base stations denoted as BS. The aim of our link model and link allocation is to provide more optimized multi-link constraints at link level by each node i, $\forall i \in$ MS. In order to facilitate the integration of four constraints, we define a set of link-level metrics, namely, (1) link access delay AD_{ik}, (2) link slot time ST_{ik}, (3) link throughput BW_{ik}, and (4) link FER ER_{ik}, as follows. If further consider an arbitrary node i, it has K_i number of one-hop neighbors within fixed transmission range, where these neighbors are k = 1, 2, ...,K_i. Meanwhile, a separate queue is attached to each mobile station for each direction of transmission. A sequence number, SN_i, is assigned to each host MS_i, by controller or the other hosts.

A. Link Access Delay

Link access delay AD_{ik} is time delay which takes to assign an optical wireless link *link(i,k)* to an access request host for transmitting data frame. The hosts that share the single optical wireless channel using TDMA method perform link access processes as follows. When a new slot time period comes up, a request host is selected to access the communication link during the slot time period.

$$SN_i \langle \left(\min_{\forall j \in MS} SN_j \right) \tag{1}$$

An access request host MS_i broadcasts request message including sequence number, SN_i which indicates its transfer order as Equation (1), and then an accessing host MS_i is determined by Round Robin method.

When the host access on the link channel, it senses the signal power level of current physical channel, determines the size of slot time ST_{ik}, and current slot time period. To share the current ST_{ik} with all the other hosts or controller, it broadcasts control frame including current slot time period. In the case of that the link access is failed, link access process can be repeatedly retried unless the current allocated slot time is not over. If AD_{ik} is less than ST_{ik} or ST_{ik} greater than 0 as Equation (2), link allocation process is repeated continuously.

$$AD_{ik} \leq ST_{ik} \tag{2}$$

B. Link Slot Time

The slot time ST_{ik} is time period which is allocated to MS_i accessing on a link channel L_{ik}. For initialization, an average slot time ST_{ik_avg} is allocated to MS_i. When MS_i accesses on the link, the size of slot time ST_{ik} is dynamically determined considering to the signal power level, LV_{ik_sig}, of physical layer that represents communication distance or signal quality between MS_i and the correspondent. The higher level of signal strength is to increase the size of slot time ST_{ik}, and the lower level of signal strength is to decrease that as Equation (3). All MS_i that share a link must share the slot time information allocated currently to MS_i accessing the link L_{ik}.

$$\begin{aligned} &Initial\_step: \\ &ST_{ik\_pre} = ST_{ik\_avg} \\ &Update\_step: \\ &Increment\_Operation: ST_{ik} = ST_{ik\_pre} + ST_{unit} \cdot LV_{ik\_sig} \\ &or \\ &Decrement\_Operation: ST_{ik} = ST_{ik\_pre} - ST_{unit} \cdot LV_{ik\_sig} \end{aligned} \tag{3}$$

C. Link Bandwidth (throughput)

The link bandwidth BW_{ik} is the volume of data frame transferred during the unit of slot time. The communication bandwidth should be maximized to maximize both link throughput and link efficiency. The size of communication bandwidth is affected by the error rate of transfer frame. As Equation (4), the higher error ratios of transfer frame should be to decrease transfer bandwidth, and the lower error ratios of transfer frame should be to increase transfer bandwidth. The MS_i that sends data frames on the link L_{ik} is to maximize the efficiency of the link communication channel by dynamically controlling the size of transfer bandwidth

$$\begin{aligned} &\mathrm{BW_{ik}} = \mathrm{BW} \cdot \log(1 + \mathrm{IF} \cdot \mathrm{SNR}) \\ &BW_{ik} \propto \frac{1}{ER_{ik}} \\ &\left(\min_{\forall j} BW_{ik}\right) \le BW_{ik\_avg} \le \left(\max_{\forall j} BW_{ik}\right) \end{aligned} \tag{4}$$

Where BW is the link bandwidth, and IF captures the unpredicted interference effects.

D. Link FER

The frame error rate depends on the quality of link channel. The communication quality of link channel is affected by signal connectivity and signal quality of physical

layer. A host configures the harmonized link channel by considering signal connectivity and signal quality of physical layer to minimize errors of link channel. It reduces frame loss and increases link channel efficiency by controlling the size of transfer bandwidth depending on the link error rate.

As Equation (5), ER_{ik} is defined as the ratio between the measured FER value and the most severe "minimum FER requirement" per link L_{ik}. We thus should select the most severe "minimum FER requirement" as the link FER constraint, and finally we have, for any successful scheduler should this outage be less or equal than one.

$$0 \le ER_{ik} \le 1 \tag{5}$$

2.4 Link Problem Definition

The physical signal characteristic of optical wireless networks shows frequent short break unlike traditional wireless signal. Therefore, it is facing the link channel problems as follows.

Firstly, a typical hierarchical structure of a wireless link model worsened the operating inefficiencies of transmission channel due to the severe discrepancy between dynamics of the transmission channel on physical layer and statics of the transmission channel on link layer. Secondly, the channel operating logic of the conventional method is simple, but its channel resource efficiency falls. Finally, as shown in Figure 1, in the case of that the existing physical layer is based on optical wireless channel, the dynamics of optical wireless channel is much more variable than that of the other wireless transmission channel. If data frame is configured with static slot on the link channel over the variable optical wireless channels, the optical wireless channel causes more serious disharmony than the existing wireless channel. It makes the efficiency of channel resource worsen. Thus, dynamic link model and failure recovery based on flexible link is needed.

To overcome the problems, an alternative must be present to give flexibility in frame structure and configuration of the link layer. Next, framing strategy should be presented to support the compatibility with the existing transport protocol hierarchy. Finally, it is required to regulate the frame size the transmission when data frame is retried due to link failure.

3 Optical Wireless Link Model

3.1 Dynamic Optical Wireless Link Model

This study communication improves link access success ratio and channel efficiency by operating dynamic link allocation scheme to enhance the link problems. It has dynamic frame configuration to cope with dynamic physical channel characteristics.

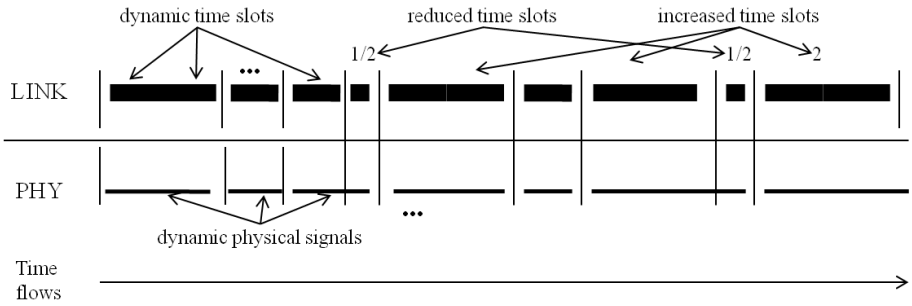


Fig. 2. Scalable Optical wireless link model

As Fig. 2, it is a scalable model that improves channel efficiency and fault recovery protocol in which is a data link layer over an optical wireless physical layer in transmission protocol stacks. Fig. 2 shows that it removes the link and signal period mismatch shown in Figure 1. This study assumes that all the sending hosts share a single transmission link to transmit data frames. The size of time slot ST_{ik} as the parameter affecting the performance of link channel allocation based on link model are adjusted in proportion to the parameter LV_{ik} indicating the level of the signal strength of the physical layer as the Equation (6).

$$ST_{ik} = \left\lceil \frac{ST_{ik\_\max} \cdot LV_{ik}}{LV_{ik\_\max}} \right\rceil \cdot ST_{unit} \tag{6}$$

And the following Equation (7) shows that the transmission bandwidth BW_{ik} of the data frame is inversely proportional to the error variable ER_{ik}.

$$BW_{ik} = \left\lceil \frac{BW_{ik\_\max}}{ER_{ik\_\max} \cdot ER_{ik}} \right\rceil \cdot BW_{unit} \tag{7}$$

3.2 Link Allocation Scheme

The size of time slot is allocated to a host sharing optical wireless link channel taking into both transmission distance and signal strength between host and host. Optical wireless terminals determine the size of time slot needs of a particular terminal by taking to the distributed control channel. For example, when a terminal requires a time slot, its slot request is accepted if it is not received reject message from the other terminals. Nevertheless, the time slot size of the optical wireless devices to share the channel will be assigned in the range of at least one third to up to three times when it is compared to the existing standard time slot by considering the tradeoff between the resource efficiency and operational complexity. The size of time slot allocation is mainly assigned of a half-size or double-size of the existing standard time slot.

```
1. program Link Allocation (BS_i, L_k);
2. const MinSigalLevels = -2, MaxSigalLevels = 2;
3. var SN_i:= 0..SeqNumofHost;
4.     SINR_ik: MinSigalLevels..MaxSigalLevels; ST_ik: 0..MaxSlotTime;
5.     ER_ik: 0..1; BW_ik: 0.. ----; End_flag:=OFF;
6. begin
7. if NewPeriod()=YES ∧ NewPeriod().SN_i =H_i.SN_i then
8.       if Request(BS_i)=ACK then
9.                 Get(SINR_ik);
10.                if SINR_ik > SINR_ik_pre then ST_ik+1;
11.                else if SINR_ik < SINR_ik_pre then ST_ik-1;
12.                Broadcast(ST_ik);
13.      if Link_Access()=YES then
14.                repeat
15.                          End_flag:=transmit_frame();
16.                          Update(ER_ik_avg);
17.                          Update(BW_ik_avg);
18.                until End_flag = YES
19.      else Retry(ST_ik-1);
20. else
21.      repeat
22.                Wait();
23.      until Finish(ST_ik)=YES
24. end.
```

Fig. 3. Dynamic link allocation algorithm

Fig. 3 shows the proposed dynamic link allocation algorithm. This link allocation scheme assumes that multiple hosts MS_i access on the shared single channel L_{ik} using TDMA-Round_Robin method. The physical signal is randomly appeared with the signal power $SINR_{ik}$. Controller C_{trn} may be a base station or each host on distributed system environments.

The initialization process of link allocation is as follows. MS_i gets its sequence number SN_i from Controller C_{trn}. MS_i gets its physical signal power $SINR_{ik}$ under current L_{ik} and it determines an average slot time ST_{ik}. The ST_{ik} may be a unit slot time. MS_i gets its error rate ER_{ik} and MS_i determines its average bandwidth BW_{ik} under current L_{ik}.

When MS_i meets its new time period and meets its conditions with sequence number SN_i, it broadcasts request packet to other hosts or controller (line 7-8). Otherwise, if the condition of new period and sequence number SN_i is not met, MS_i waits until it meets the next slot time period (line 20-23).

If MS_i gets acknowledgment from controller C_{trn}, it gets its physical signal power $SINR_{ik}$ under current L_{ik} (line 9), and it increases ST_{ik} if $SINR_{ik}$ is greater than $SINR_{ik\_pre}$, and it reduces ST_{ik} if $SINR_{ik}$ is less than $SINR_{ik\_pre}$ (line 10-11). Then MS_i broadcasts its slot time ST_{ik} to other hosts or controller (line 12). If the access of MS_i on the current link L_{ik} is successful, MS_i transfers data frames on the current link (line 13-15), and MS_i updates its error rate ER_{ik} and its average bandwidth BW_{ik} under current L_{ik} (line 16-17). MS_i repeats data transfer until it is finished (line 18). If the link access of MS_i is failed, it retries with minimized its slot time ST_{ik} to access on the current link until it is over its maximum slot time size (line 19).

3.3 Link Allocation Criterion

Link allocation process is performed by separating the active and inactive terminals depending on the link connection state of a terminal.

The link process of active terminals is processed as the activation initialization, the access control frames transmission, the data frames transmission, and the link disconnection. The initialization checks the new slot period of link channel, and determines the accessing terminal and the time slot size. Then, the access terminal sends an access control frame. If it is success, transmit data frames. Otherwise, reduce the size of current slot time after checking the signal power of physical link time. The active terminal assigned with time slot on the link channel transmits data frames. If it was successful (ACK), send the next data frame repeat to the next slotted time period. If it was failed (NAK), retry to send the data frame after checking the signal power of physical link and adjusting the size of current slotted time. Finally, if an active terminal completes the data transmission or terminates the slot period, its status is switched into inactive status and the link medium is converted to idle status.

Inactive terminal checks the current time slot period. It waits for the next time slot period. It requests the access of link channel in each new slot period. If the request is approved, it is converted to the active terminal. Otherwise, it waits for until its request is accepted.

4 System Analysis and Simulation

4.1 Experimental Environments

This work assumes that analysis model has three (3) level hierarchy architecture of base station (BS) as a network coordinator, access point (AP) and wireless terminal for transmission efficiency improvement. The requested data of wireless client are transferred from BS to AP, from AP to MC and vice versa.

An optical wireless link allocation is performed through one access point in the shared optical wireless link over distributed systems. This paper assumes that all mobile hosts just use a shared one optical wireless link.

The base station coordinates the messages among assess points, and wireless terminal accesses to optical wireless media link using TDMA protocol. The wireless terminal and optical wireless links will change randomly. It is assumed that all terminals have standard time slot and know a period of time slot for current allocated channel.

4.2 System Analysis and Evaluation

Existing wireless link model, as shown in Figure 1, showed the period mismatch model between the signal connection of optical wireless physical layer and the slot period of link layer. This can degrade the frame transmission performance and resource efficiency. The proposed optical wireless link model enhances the transmission capability by removing mismatched links, which were appeared in the existing static links model, using the dynamic link. At the same time, it is to maximize the link channel efficiency by adjusting the size of link channel depending on the link error

rate. This section compares the performance and the resource efficiency for each conventional link model and proposed model through experiments. First, it shows the link scaling information according to the level of the signal strength of the physical layer for each of the existing single mode and the proposed multiple mode. Second, when a wireless host gets a time slot and tries to access the link channel, it compares the success ratios for each the existing model and the proposed model. Third, when hosts continuously access to link, it compares the link efficiencies of each existing model and proposed model that represents the error recovery capability.

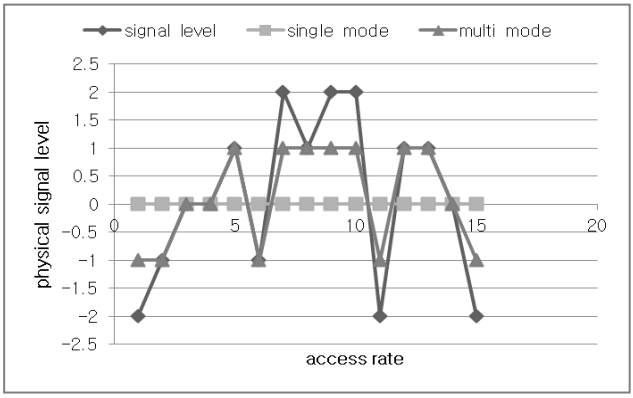


Fig. 4. Physical signal vs. Link modes

Fig. 4 shows the differential gaps on single or multiple modes compared to five signal levels by each access request of mobile host when physical signal power is randomly changed. 0-level displays average signal power. Plus-level ("1, 2, 3") values mean that access hosts have higher signal power than average 0-level value. Minus-level ("-3, -2, -1") values mean that they have lower signal power than it.

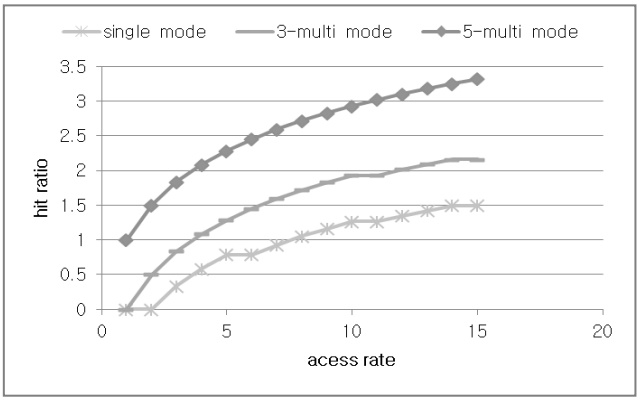


Fig. 5. Access hit ratio over access call rate

Fig. 5 shows relative hit ratios by continuous access requests of mobile hosts. Previous model with single mode has lower hit ratios compared to that of the proposed model with three link modes ("-1, 0, 1") and five link modes ("-2, -1, 0, 1, 2"). Here, the hit ratio of the proposed model with five link modes displays higher values than that of the proposed model with three link modes.

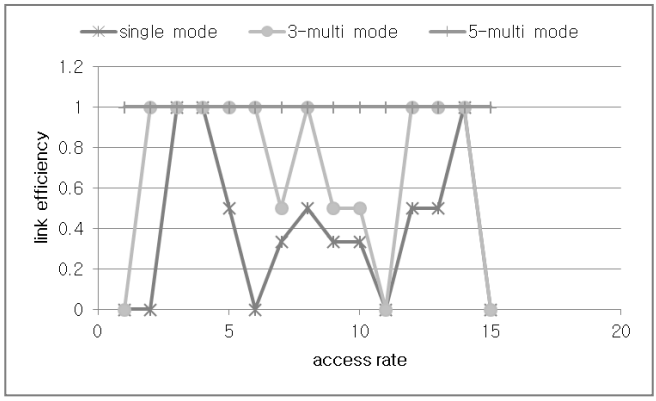


Fig. 6. Link efficiency over access call rate

Fig. 6 shows link efficiencies of mobile hosts with different link modes. Previous model with single mode has lower link efficiency compared to the proposed model of three link modes ("-1, 0, 1") and five link modes ("-2, -1, 0, 1, 2"). Especially, the five-link mode ("5-multi mode") graph shows perfect link efficiency under signal classification with five link levels.

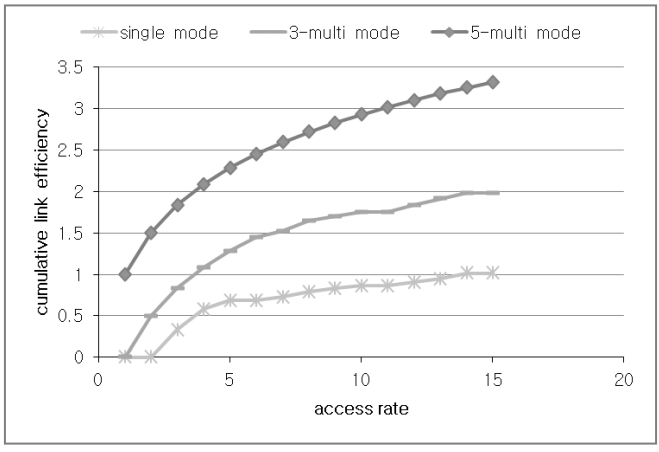


Fig. 7. Cumulative link efficiency over access call rate

Fig. 7 shows link efficiencies by continuous access requests of mobile hosts. Previous model with single mode has lower efficiencies compared to that of the proposed model with three link modes ("-1, 0, 1") and five link modes ("-2, -1, 0, 1, 2"). Here, the efficiency of the proposed model with five link modes shows higher values than that of the proposed model with three link modes.

5 Conclusion

As shown in analysis above, the proposed agent transfer methods produce clear advantages in terms of packet loss ratios and data propagation delays.

This paper has main contributions that enforce transfer performance, resource efficiency and fault-recovery performance. First, in Sections 4.2, the transmission performance of the proposed model was better than the existing model. The proposed model has greatly improved setup time delay as well as transmission delay. Second, it has improved the utilization and efficiency of the channel resources by enabling the discarded link channel of the existing model as shown in Sections 4.2. Finally, as shown in Sections 4.2, it also has improved the success rate of frame transmission by adjusting the size of time slot at the time of frame retransmission.

Future research in the dynamic link allocation environments will need the load balancing method for avoiding overload, and the QoS link management method for providing the priority transmission services.

References

1. Williams, S.: IrDA: Past, Present and Future. IEEE Personal Communications 7, 11–19 (2000)
2. Axtman, D., Ogus, A., Reilly, J.: Infrared Data Association LAN Access Extensions for Link Management Protocol, IRLAN (Extended Systems Incorporated, Hewlett-Packard Corporation and Microsoft Corporation, July 18 1997), IrDA Version 1.0
3. Clark, P., Sengers, A.: Wireless optical networking challenges and solutions. In: IEEE Military Communications Conference, MILCOM, CA, USA, pp. 416–422 (October 2004)
4. Bouchet, O., O'Brien, D., Tabach, M.E., Mach, N., Faulkner, G., Rosal, L.F., Franke, M., Grubor, J., Kamalakis, T., Langer, K.D., Minh, H.L., Neokosmidis, I., Nerreter, S., Ntogari, G., Walewski, J.W., Wolf, M.: Seventh framework programme Theme 3 Information and Communication Technologies (ICT), State of the Art – optical wireless, ICT-213311 OMEGA, Deliverable D4.1 (2008), http://www.ict-omega.eu
5. Vitsas, V., Barker, P., Boucouvalas, A.C.: IrDA infrared wireless communications: Protocol throughput optimization. IEEE Wireless Communications 10(2), 22–29 (2003)
6. Cao, G.: Designing Efficient Fault-Tolerant Systems on Wireless Networks. Cert Coordination Center, 2000. Carnegie Mellon Software Engineering Institute (December 2001), http://www.cert.org/research/isw/isw2000/papers/39.pdf
7. Cuiñas, I., Martínez, D., Sánchez, M.G., Alejos, A.V.: Modeling and Measuring Reflection Due to Flat Dielectric Surfaces at 5.8 GHz. IEEE Transactions on Antennas and Propagation 55(4) (April 2007)

8. 802.11-1997 - IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, pp. 1-445 (June 1997)
9. Chatzimisios, P., Vitsas, V., Boucouvalas, A.C., Kleftouris, D., Tsoulfa, M.: Packet Delay Performance Comparison of the IEEE 802.11 and IRDA AIR CSMA/CA Protocols in High-speed Wireless LANS. In: Proceedings of the 24th IASTED International Multi-Conference Internet and Multimedia Systems and Applications, Innsbruck, Austria, February 13-15, pp. 171–176 (2006)
10. Bolton, P.R., Deprey, C.L.: IR Bandwidth and Crystal Thickness Effects on THG Efficiency and Temporal Shaping of Quasi-rectangular UV pulses: Part II-Incident IR Intensity Ripple. LCLS-TN-07-4, June 01 (2007)
11. Kahn, J.M., Barry, J.R.: Wireless Infrared Communications. Proceedings of the IEEE 85(2), 265–298 (1997)
12. Mathew, A., Chandrababu, N., Elleithy, K., Rizvi, S.: Interference of 802.11b WLAN AND Bluetooth: Analysis and Performance Evaluation. International Journal of Computer Networks & Communications (IJCNC) 2(3), 140–150 (2010)
13. Zussman, G., Segall, A., Yechiali, U.: Bluetooth Time Division Duplex-Analysis as a Polling System. In: First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, pp. 547–556. IEEE SECON, Santa Clara (2004)

Addressing Situational Awareness in Critical Domains of a Smart Grid

Cristina Alcaraz and Javier Lopez

Computer Science Department - University of Malaga,
29071 - Malaga, Spain
{alcaraz,jlm}@lcc.uma.es

Abstract. Control and situational awareness are two very important aspects within critical control systems, since potential faults or anomalous behaviors could lead to serious consequences by hiding the real status of supervised critical infrastructures. Examples of these infrastructures are energy generation, transmission or distribution systems that belong to Smart Grid systems. Given the importance of these systems for social welfare and its economy, a situational awareness-based model, composed of a set of current technologies, is proposed in this paper. The model focuses on addressing and offering a set of minimum services for protection, such as prevention, detection, response, self-evaluation and maintenance, thereby providing a desirable protection in unplanned situations.

Keywords: Critical Infrastructure Protection, Smart Grid, Supervisory Control and Data Acquisition Systems, Situational Awareness, and Wireless Sensor Networks.

1 Introduction

A Smart Grid is a complex infrastructure composed of a set of domains and stakeholders. According to the conceptual model of the National Institute of Standards and Technology (NIST), these domains correspond to customers, markets, providers, energy generation, distribution and transmission networks (e.g., power substations), as well as control systems such as SCADA (Supervisory Control and Data Acquisition) systems [1]. This last domain can be considered as the main core of the entire system that widely interconnects with the other domains/sub-domains. This interconnection enables the SCADA Center to know the performance of the entire Grid and control its functions for delivering essential services, such as electrical energy.

Unfortunately, control substations in charge of supervising in real-time the performance and functionality of energy bulk generation systems (either renewable or non-renewable), or electrical transmission or distribution lines have a tendency to suffer numerous and unforeseen events caused by failures or errors. The origin of these suspicious events may even provoke disturbances or instabilities within a particular substation that could trigger a devastating cascading effect, with a high probability of reaching other domains within the Grid. This is due to the existing interdependency relationships [2, 3] that may intensify the spread of the effect, thereby (partially or totally) disrupting functionalities/services of other domains/sub-domains.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 58–71, 2012.

We agree with NIST that it is necessary to provide preventive and proactive solutions to face emergency situations [1]. In fact, NIST classifies this need as one of the eight priority areas to be considered for the protection of Critical Infrastructures (CIs), and it is known as Wide-Area Situational Awareness (WASA). Given its importance within a Smart Grid, in this paper we propose a model based on the use of different technologies to ensure control at all times, in addition to offering a support for situational awareness. The proposed approach is specifically composed of:

- The technology of Wireless Sensor Networks (WSNs) for monitoring the actual state of the infrastructure observed and its industrial resources (e.g., turbines);
- The ISA100.11a standard [4] for managing different kinds of SCADA incidents, represented through alarms and classified into five levels of priority;
- Two preventive methods. One of them focusing on anticipating critical situations and the other on controlling anomalies or malfunctions in the control tasks.
- Cloud computing based on Sensitive Data (SD) for data redundancy (i.e., alarms and readings) and safety-critical; i.e., take control of a highly critical situation to avoid the propagation of a cascading effect [5]; and
- A self-validation mechanism to evaluate the real state of the entire system, itself.

Self-validation basically consists of evaluating the level of accuracy of the methods applied for protection of CIs. These methods correspond to the prevention (anomalous situations related to the infrastructure controlled) and/or detection (anomalies or threats within the control network). This detection is mainly based on the use of simple behavior patterns that help to detect unsuitable (hardware and software) functions in sensor nodes, thereby offering a support for maintenance and auditing tasks. Note that some of these solutions try to address new research areas, such as cloud computing for critical contexts, and others try to fill some research gaps such as prevention. Indeed, although there are some action plans and initiatives [6] to provide preventive solutions, there is not so far enough research on this topic for critical contexts; and more particularly in the provision of specialized predictive solutions based on simple forecast models.

The paper is organized as follows. Section 2 introduces the basic components for the construction of the approach, which will be later used for the design in Section 3. In particular, the approach and its components, technologies and methods for prevention, detection, response and self-validation are discussed in detail in Section 3.1 and Section 3.2. Finally, Section 4 concludes the paper and outlines future work.

2 Four Basic Components for the Construction of the Approach

A system based on situational awareness basically comprises advanced monitoring components with integrated techniques that help to analyze and interpret data streams, normally from embedded devices (e.g., sensor nodes), which are distributed close to the controlled infrastructure (e.g., machineries). Likewise, these techniques have the capability for decision-making and alerting. Therefore, four main components should form the foundations of our approach; (i) a *detection component*, (ii) an *information recollection component* to store evidence, (iii) an *alarm management component* to issue alerts and warn the system, and (iv) a *reaction component*. The detection component is based

on WSNs since their devices (sensor nodes) are able to monitor physical events (such as high/low levels of voltage); detect and track anomalous behaviors; warn of anomalous situations; and actively interact with the gateway [7]. The gateway is a powerful device that serves as an interface between the acquisition world (i.e., the WSN) and the real world (i.e., the SCADA Center). In addition, these sensor nodes are smart devices with the capability of collaborating with each other and guaranteeing self-configuration to adapt themselves to the conditions of the network, as well as self-healing to address unforeseen situations.

The information recollection component in our model is represented by the SCADA Center itself, the SD cloud and any external storage device in charge of registering and storing SCADA evidence flows. The use of cloud computing for evidence storage enables the system to maintain a register of events occurred in the past. If the control is (temporarily or permanently) lost (e.g., the SCADA Center is out of service), another SCADA system may retake control through the ICCP (Inter-Control Center Communications Protocol) industrial protocol, and know the state of the system by querying the DS cloud [5]. The effectiveness of using this technology and its application for managing incidents in critical contexts are thoroughly analyzed in [5]. In fact, one of its great advantages is the availability of resources, keeping control at all times and recovering sensitive data irrespective of the situation; and in this way ensuring a continued supervision and safety-critical in crisis scenarios. A safety-critical is considered an essential property [3] that should be considered when the underlying infrastructure is critical, as the existence of unplanned events may potentially lead to serious consequences [2, 3]; e.g., overload in generators, high voltage peak in transformers, etc.

The alarm management component is based on specific management systems offered by existing wireless industrial communication standards, such as ISA100.11a. This standard provides a set of services for communication reliability, security (based on symmetric and asymmetric cryptography), coexistence, and priority-based alarm management using up to five criticality levels: *journal, low, medium, high* and *urgent*. Its networks can support sensor nodes working at 26MHz, 96KB RAM, 128KB flash memory and 80KB ROM, and one or several gateways to establish redundant connections with the SCADA Center. The information from sensors is managed through DMAP (Device Management Application Process) objects. DMAP is a class installed inside each device, which includes a set of objects used for configuring, supervising and requesting parameters belonging to sensor nodes. More specifically, DMAP contemplates the ARMO (Alert Reporting Management Object) class for managing alerts and generating reports through an AlertReport service to ARO (Alert Receiving Object). ARO is a class configured in only one device in the network (the gateway in our case). Finally, the reaction component focuses on carrying out decision-making processes that depend on a set of factors, amongst others, the simplicity of the technique applied (which should not increase functional complexities that can compromise the control of the underlying infrastructure and its services) and the autonomous and dynamic capacity of the approach to address threatening situations. In our case, this component is principally based on a set of integrated modules that collaborate with each other to carry out several tasks. Some of them are; to estimate the proximity of a possible anomaly; locate

and warn the nearest operator in the area; evaluate the level of accuracy in the detection and prevention tasks; and frequently report the real state of the network.

3 A Dynamic and Automatic Situational Awareness

As ISA100.11a allows configuring diverse types of networks, the architecture of the approach (See Fig. 1) is based on a hierarchical configuration; where nodes are grouped into clusters and all the organizational decisions are carried out by a trustworthy entity known as the *Cluster Head* (CH). Each CH_i is responsible for receiving and checking information (either readings or ISA100.11a alarms) from their sensors in order to detect and warn of anomalous behaviors through patterns, in addition to filtering and aggregating information (main tasks of a CH) to be resent to the gateway later. The selection of this configuration is for two main reasons. First of all, this configuration not only allows the system to efficiently manage its resources in computation and energy terms, but it also helps to locate anomalies by knowing the network deployment in advance. Second, part of the processing is straightforward, since the approach has been designed for very specific situations using simple behavior patterns.

An anomalous behavior can be defined as "*something deviated from what is standard, normal, or expected*". From this definition, taken from the Oxford Dictionary [8], we deduce that if a reading is not inside a prescribed threshold, $[V_{min}, V_{max}]$, then it can be considered anomalous. As our approach measures readings of voltage, denoted as v_i, a deviation from the allowable thresholds is therefore considered as an anomaly. When this situation appears, the system has to deliver an alarm. Taking advantage of ISA100.11a and its alarm management, we can consider three principal situations:

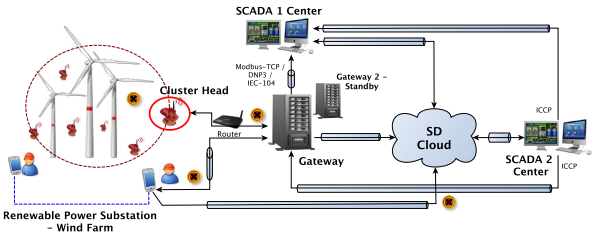


Fig. 1. General Architecture of the Model

- Valid readings, $v_i \in [V_{min}, V_{max}]$, where V_{min} and V_{max} refer to the acceptable thresholds of readings. To highlight and signal this case, we use the value 0.
- Non-critical alarms, $v_i \notin [V_{min}, V_{max}]$, but they do not compromise the security of the system. These alarms are journal, low and medium, and are signaled with values 1, 2 and 3 respectively.
- Critical alarms, $v_i \notin [V_{min}, V_{max}]$, but they can compromise the security of the system. These correspond to alarms with high and urgent priority, which are signaled with values 4 and 5 respectively.

The gateway is in charge of resending any type of information (valid readings, non-critical alarms and critical alarms) from the WNS to the SCADA Center; interpreting and translating (e.g., Modbus-TCP/IP - ISA100.11a) messages using GSAP (Gateway Service Access Point) points; and storing information copies in the SD cloud for backup. It is also responsible for anticipating future anomalies, managing critical alerts [4-5], and validating the entire approach itself. For dealing with critical alerts, the gateway also has to locate the most suitable operator equipped with a hand-held device within the area, which makes use of different communication systems (e.g., Mobile Ad-Hoc Networks). On the other hand, although security aspects are beyond the scope of this paper, we assume that communication channels 'sensor-sensor' are protected by using security credentials and cryptographic services provided by the ISA100.11a standard [4]; and the rest of the communications will depend on the security services of the TCP/IP standard and on the use of virtual private networks.

3.1 Sensors and The Cluster Head for Dissemination and Detection

Figure 2 depicts the chief modules of the CHs: *Message Normalization*, *Pattern Association*, *Alarm Manager* (AM-CH), *Data Aggregation*, and *Diagnosis Manager*. Each sensor node, s_i, with identification IDs_i sends its messages (either a v_i or an alarm) to its CH$_j$ with IDch_j, which first operates the Message Normalization module. The main task of this module is to combine and represent different data inputs in a generic format. The normalized message is then sent to the Pattern Association module in order to verify the nature of such inputs using simple behavior patterns. For example, verify if readings or critical alarms received from a s_i are outside their acceptable thresholds before being forwarded to the gateway. In this way, we can make good use of the cluster head by supervising the functional instabilities of the nodes included within it. These instabilities may be, for example, caused by software/hardware errors or malfunctions due to a lack of maintenance. Depending on the detected anomaly, the AM-CH module will generate, through the ARMO class, a new alarm signaled with high priority (4) so that a human operator can be made aware of the situation and can review the scenario.

For simplicity, we consider the following network model. The network deployment is based on trustworthy nodes where sensors are distributed close to their cluster heads, and each cluster is based on a small configuration of nodes. Each node has to transmit a message with the value of the reading and priority assigned, the identifier IDs_i and the time-stamp. To address the software malfunction problems, each CH must verify the payload of each message to check whether its value of reading corresponds to the priority assigned by the sensor; e.g., verify whether $v_i \in$ (or $\notin$) $[V_{Low_{min}}, V_{Low_{max}}]$. These thresholds of criticality must be defined according to security policies established by the SCADA organization, electrical companies and countries. Only in the case where a CH analyzes a discrepancy in the control made by a sensor, the CH then has to penalize its attitude by updating its behavior counter, $counter_{SensorBh}$ by one unit. This counter is unique for each node and when its value is greater than a prescribed behavior threshold (i.e., $counter_{SensorBh} > T_{SensorBh}$), the CH will also have to warn the AM-CH.

On the other hand, hardware problems are managed using the Diagnosis Manager, which periodically queries the last sequence of events received from the sensors using a cache memory. This memory, which is maintained by the Message Normalization,

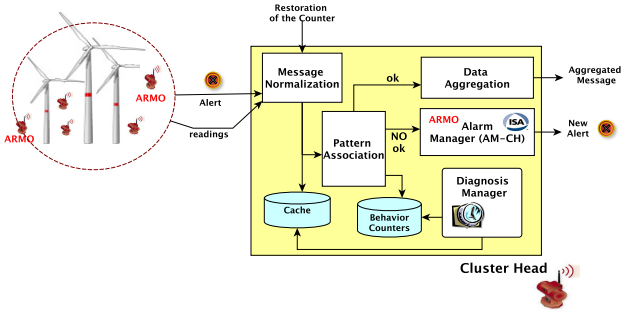


Fig. 2. Architecture of the Cluster Head

allows the Diagnosis Manager to know when a particular node of the cluster is not sending messages for a short time period. If this occurs, the CH infers that something anomalous is happening with the sensor, and updates its $counter_{SensorBh}$. This problem could be attributed to a significant reduction in battery levels or the lifetime of the sensor is over. It should be noted that the counter used coincides with the behavior counter described above, because when a node is behaving incorrectly, the system increases (without any distinction of the cause) its value until it reaches its threshold, $T_{SensorBh}$. In that moment, the CH will have to warn of the situation so that the sensor can be tested.

For generating a new alarm, both the Pattern Association and the Diagnosis manager will have to send the AM-CH a set of data. For example, the IDch_j; IDs_i; the type of alarm (only if the received message from s_i is an alarm); the priority assigned by the sensor; the priority assigned by the CH; and the type of event detected. The kind of event is an indicator that will help to make the gateway and the human operator aware of the type of problem to check. It should be noted that this type of validation is only effective for critical alarms [4-5], since valid readings and non-critical alarms will be used as input for prevention. In particular, two types of events are used: *event_detectionSensor* and *event_detectionCH*. The former refers to the detection made by a sensor node (i.e., the control of the CI and its services), whereas the latter is attributed to the detection carried out by the CH (i.e., the control of behaviors within the cluster). To show the simplicity of the Pattern Association module, the Pseudo-Code 1 summarizes the order of execution of its actions.

We have validated this part of the approach using the Avrora simulator under the de-facto standard operating system for sensor nodes, TinyOS 2.x [9]. Avrora is able to interpret conventional sensor nodes (e.g., Mica2), which belong to the category II defined in [7]; i.e., 4-8 MHz, 4-10 KB RAM, 48-128 KB ROM with 2-8 mA of energy. The results of the simulation (See Table 1) indicates that a cluster working as a Mica2, requires less than 8MHz to execute the software, consuming around 3,3 Joule for CPU and 8.6 Joule for radio, and approximately reaching a maximum of 2.8% for reading (r) and a 3% for writing (w) in memory. Therefore, if traditional sensors are able to work as CHs, then ISA100.11a sensors belonging to the category III with higher capabilities (13-180 MHz, 256-512 KB RAM, 4-32 MB ROM and 40 mA of energy) are also able to server as CHs.

```
//Obtain normalized message and extract values to analyze
message = NormalizedMessage();
reading = Extract_ReadingData(message);
prioritySensor = Extract_Priority(message);
IDs_i = Extract_IdentifierSensor(message);
//Verify the accuracy of the sensor to assign priority
IF (VerifyData(reading,prioritySensor)) THEN
        IF (Priority(prioritySensor, 0)) THEN
                //Aggregate whether the contain is a reading
                DataAggregation(IDs_i, reading);
        ELSE
                //Resend the alarm to the Gateway
                ForwardAlarm_AM-CH(IDch_j, IDs_i, reading, prioritySensor, "event_detectionSensor");
        END
ELSE
        //Determine the real criticality of the received reading according to behavior patterns; and
        //Update the counter_SensorBh of the sensor node s_i
        priorityCH = DeterminePriority(reading);
        counter_SensorBh_i = UpdateBehaviorCounter(IDs_i);
        IF (counter_SensorBh_i ≤ T_SensorBh) THEN
                //Generate a new alarm to evaluate behaviour in the gateway
                GenerateNewAlarm_AM-CH(IDch_j, IDs_i, high, prioritySensor, priorityCH, "event_detectionCH");
        ELSE
                //Generate a new alarm to warn the operator of the replace/discard of the sensor
                GenerateNewAlarm_AM-CH(IDch_j, IDs_i, high, "event_discardNode");
        END
END
```

Pseudo-Code 1: Control of Software/Hardware Malfunctions within a Cluster

Table 1. Resources of one Cluster Head When Sensors Are Being Integrated within the Cluster

Resources	1 CH - 0 sensors	1 CH - 1 sensor	1 CH - 2 sensors	1 CH - 3 sensors
CPU	7,36MHz	7,37MHz	7,37MHz	7,36MHz
Memory (r-w)	2,75% - 3,02%	2,74% - 3,01%	2,73% - 2,99%	2,72% - 2,98%
Energy (CPU-Radio)	3,31 J - 8,62 J	3,31 J - 8,61 J	3,31 J - 8,62 J	3,32 J - 8,63 J

3.2 A Powerful Gateway for Control, Prevention, Response and Maintenance

As part of the approach, a gateway is integrated inside the model (See Fig. 3), which is composed of two chief managers: An *Incident Manager* and a *Maintenance Manager*.

Incident Manager: Prevention, Data Redundancy and Response. Any type of information received from CHs is taken in through the *ARO* sub-module, which temporarily stores them within a cache memory and send a copy to both the SCADA Center and the SD cloud. For incident management, ARO uses one organized queue, which is sorted by priorities. Depending on the criticality of the message, the *Alarm Manager* (AM-GW) sub-module will carry out two actions; one predictive and other reactive. For the predictive part, the AM-GW must compute the rate of valid readings (0) and non-critical alarms [1-3] received from the network. The idea is to calculate, for each sensor, rates of consecutive values of non-critical alarms with value 3 over the last time period, as it may mean the proximity of a possible incident. Although, there are currently several forecast models that could be used to anticipate such situations [10], we propose below a simple prevention method, which is included inside the *Prediction* sub-module belonging to the AM-GW.

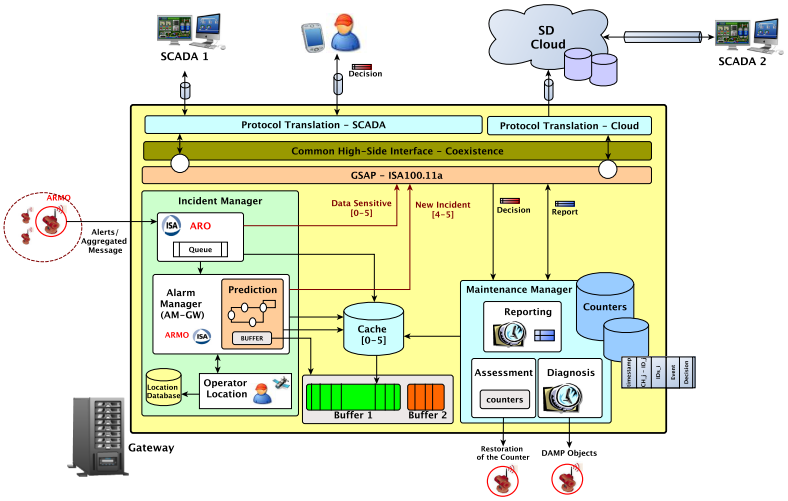


Fig. 3. Architecture of the ISA100.11a Gateway

The method basically consists of calculating probabilities of transition between states: st_0 (represents valid readings), st_1, st_2, st_3 (represents different types of criticality [1-3]). These states and their values have to be previously exported from the cache memory to a separate temporal buffer, which is assigned to each network sensor, Bff_i, with a size Δ_{Bff_i}. However, this buffer is not only based on information exported from the cache, but also on past information (a small percentage) in order to keep a sequence of events with respect to the time line. Therefore, the size of Bff_i is based on exported information with a size of Δ_{Bff1_i} and on past information with a size of Δ_{Bff2_i}; i.e., $\Delta_{Bff} = \Delta_{Bff1} + \Delta_{Bff2}$, where $\Delta_{Bff1} \geq \Delta_{Bff2}$. In this way, we can restrict the size of Δ_{Bff} and reduce computational costs by avoiding to computing several times the predictive algorithm and the cache memory.

For each of the states, we also design a particular probability of transition pr_{st_α, st_β}, which corresponds to the probability of going from a state α to a state β; i.e., $pr_{st_\alpha, st_\beta} = Pr(st_{i+1} = \beta | st_i = \alpha)$, where $\sum_{i=0}^{3} pr_{st_\alpha, st_\beta} = 1$. Taking this into account, we assume that the probability of remaining in the st_0 is much greater than transiting to the st_3 or remaining within this; i.e., $pr_{st_0} > pr_{st_1} > pr_{st_2} > pr_{st_3}$. In order to calculate probabilities, we consider the following Equation: $1/(4 \times \alpha)$, where $\alpha >= 1$ and $pr_{st_0} = 1 - (\sum_{\alpha=1}^{3} pr_{st_\alpha})$. Note that we have taken this simple equation as an initial approach. Other approaches could also be equally valid if they are achieved with the restriction of $pr_{st_0} > pr_{st_1} > pr_{st_2} > pr_{st_3}$. The result of computing the probabilities for each state is as follows: pr_{st_0} - 0.542; pr_{st_1} - 0.25; pr_{st_2} - 0.125; and pr_{st_3} - 0.083. Figure 4 graphically depicts the relationships between states together with the cost of their transitions.

Considering the previous assumptions and notions, the occurrence of an event can be computed as follows.

$$\frac{InitialState + \sum_{j=0}^{\Delta_{Bff_i}-1} pr_{Bff_i[j], Bff_i[j+1]}}{\Delta_{Bff_i}} \leq (pr_{st_3} + \sigma_{error}) \quad (1)$$

where *InitialState* corresponds to $pr_{Bff_i[0]}$ and σ_{error} represents an acceptable margin of error. This means that if the result of computing Equation 1 is lower than $pr_{st_3} + \sigma_{error}$, the system can determine that the next value to be received will be either a non-critical alarm with value 3 or a critical alarm (a stressed situation). To the contrary, when the system determines that the result of computing Equation 1 is higher than $pr_{st_3} + \sigma_{error}$, it may infer that the next entry may be either a valid reading or a non-critical alarm (a normal/acceptable situation). To make this clearer, two examples are shown below, which are based on a $\Delta_{Bff_i} = 10$ ($\Delta_{Bff1_i} = 5$ and $\Delta_{Bff2_i} = 5$) with a $\sigma_{error} = 0$.

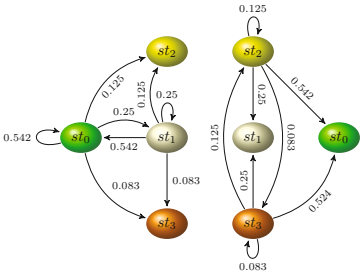


Fig. 4. Transitions Between States: From st_α to st_β

- Let the sequence of events stored in a Bff_i as 0 3 2 3 3 3 0 3 3 3, the system then computes the transitions and their probabilities using Equation 1. Resulting in, $0.179 > pr_{st_3}$. Then the system estimates that the next event to be received may be either a valid reading or a non-critical alarm.
- If the sequence of events has 3 3 3 3 3 3 3 3 3 3, the result of calculating would be $0.083 \leq pr_{st_3}$. Therefore, the system determines that the next event to be received may be either a non-critical alarm with value 3 or a critical alarm due to the high rate of alarms received with value 3.

To address this last case, and of course the reactive part, the system has to warn of the proximity of this situation by sending a new alarm with high priority through the AM-GW. Such an alarm must be sent to both the SCADA Centre and the nearest operator within the affected area so as to immediately attend to the situation. Similarly this can also occur when ARO directly receives critical alarms [4-5] from the sensor network (e.g., alarms with the type of event "*event_discardNode*"). For operator location, the AM-GW uses the *Operator Location* sub-module, which considers the operator's availability (according to his/her contract), his/her responsibility/role to carry out a task, and his/her location within the area. To carry out such a search, the Operator Location makes use of both a local database, called *Location Database*, and a location external device, such as a geospatial information device, so as to geographically identify the physical position of the nearest human operator within the affected area. Lastly, and as mentioned in Section 2, the AM-GW not only has to send a copy of new incident generated to the SCADA Center but also to the SD cloud for future governance aspects and recovery purposes.

Maintenance Manager: Self-validation and Maintenance. In order to know the real state of the entire approach, the *Assessment* sub-module needs to receive certain feedback on how accurate the prevention and detection modules have been. This feedback is dependent on the operator's final decision, who is obliged to verify, validate and notify (through their hand-held interfaces) the system of the reliability of the detection/prevention made by the control network. In fact, four possible situations could occur: (i) The node determines that an anomaly is occurring within the system, and it coincides with the operator's decision (a True Positive (TP)); (ii) the node determines that an anomaly is occurring within the system, and it does not coincide with the operator's decision (a False Positive (FP)); (iii) the node determines that no anomaly is occurring within the system, and it does not coincide with the operator's decision (a False Negative (FN)); and (iv) the node determines that no anomaly is occurring within the system, and it coincides with the operator's decision (a True Negative (TN)). It should be noted that a TN does not make sense within our approach, thus it has not been considered.

Table 2. Table for Evaluating the Prevention and Detection Modules, and Updating Counters

	Prevention	**Detection and Control of the CI**	**Detection and Control of the Cluster**					
	priority	***prioritySensor***	***priorityCH***			***prioritySensor***		
prirotyOp.	**High**	**High**	**0**	**[1-3]**	**[4-5]**	**0**	**[1-3]**	**[4-5]**
Normal Sit. - 0	FP	FP	TP	FP	FP	TP*	FP	FP
Unstable Sit. - [1-3]	FP	FP	FN	TP	FP	FN	TP*	FP
Critical Sit. - [4-5]	TP	TP	FN	FN	TP	FN	FN	TP*

Depending on the operator's decision, the Assessment sub-module will have to update the level of accuracy for a node using three kinds of counters (associated with each network node); $count_{tp}$ for TPs, $count_{fp}$ for FPs, and $count_{fn}$ for FNs. If said counters reach their respective prescribed thresholds, then the Assessment sub-module will have to issue a new alarm with a high priority through the AM-GW. The new alarm should contain, at the very least, information related to the nodes involved (e.g., IDs_i, IDch_j, IDgw) and the action to be carried out, such as *event_review_detectionModule*, *event_review_predictionModule*, or even *event_discardNode* (discard/replace devices).

For evaluating the prevention, it is enough to take into account the operator's decision and the estimation of the Prevention sub-module. The operators' decision is going to depend on three types of criticality levels: *normal situation* (0), *unstable situation* [1-3], and *critical situation* [4-5]. For example, if the operator's feedback corresponds to a *normal situation/unstable situation* (See Table 2), the $count_{fp}$ of the Prevention sub-module should be increased accordingly. This validation method is equivalent to evaluate the reliability of sensors in their control tasks of CIs (with event *event_detectionSensor*, See Section 3.1); and the reliability of CHs in their supervision tasks of malfunctions (with event *event_detectionCH*, See Section 3.1). Nonetheless, it is worth mentioning that this last kind of validation is a little more complex, as the sub-module requires contrasting the version of the CH$_j$ (i.e., *priorityCH*, See Pseudo-Code 1 of Section 3.1) and the version of the sensor involved, IDs_i, (i.e., *prioritySensor*, See Pseudo-Code 1 of Section 3.1) with respect to the criticality provided by the human operator (i.e., *priorityOp*). When contrasting versions, a further two specific situations may take place:

- The *priorityOp* coincides with the *priorityCH*; i.e., TP in CH: The system rewards the CH by increasing its $count_{tp}$, and penalizes the s_i according to the real criticality of the system. Hence, if $priorityOp > prioritySensor$, then $count_{fn}$ of the s_i is increased; otherwise, its $count_{fp}$ is updated by one unit.
- The *priorityOp* does not coincide with the *priorityCH*; i.e., FP/FN in CH: The system increases the $count_{fp}$/$count_{fn}$ of the CH, accordingly. However, a further two cases may also occur when the the counters of the sensor have to be updated:
  - The *priorityOp* is equal to the *prioritySensor*; i.e., TP in s_i: The system rewards the s_i by updating its $count_{tp}$, and proceeds to restore the value of the $counter_{SensorBh_i}$ (See Section 3.1). To this end, the Assessment sub-module has to send a notification to its corresponding CH_j to increase its value by one unit. Note that this action, also depicted in Table 2 using the indicator '*', significantly reduces the communication overhead. If this counter was managed by the gateway, this could mean a high communication cost, as the $counter_{SensorBh}$ needs to be continuously updated by the Association Pattern module and Diagnosis Manager of the CH.
  - The *priorityOp* is not equal to the *prioritySensor*; i.e., FP/FN in s_i: If the *priorityOp* is less than the *prioritySensor*, the system increases the $count_{fp}$ of the s_i; otherwise the system penalizes the node by increasing its $count_{fn}$.

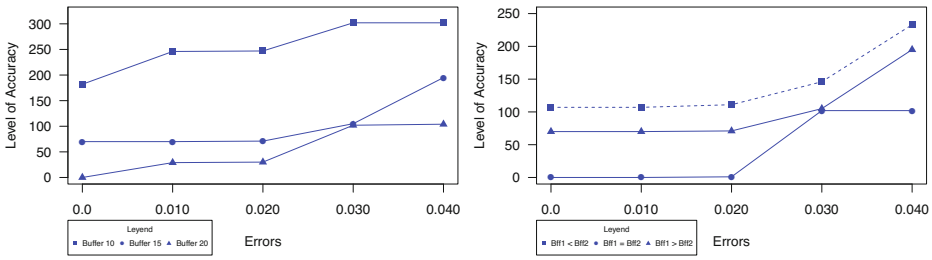


Fig. 5. Left Hand Side Figure: The Importance of σ_{error} and Δ_{Bff} for Critical Contexts; Right Hand Side Figure: The Importance of Δ_{Bff1} and Δ_{Bff2}

When a $count_{fp}$ and/or a $count_{fn}$ reach their acceptable thresholds (T_{fp} and T_{fn}, respectively), the SCADA Center should be warned in order to take new protection and security measures, and thereby guarantee continuity of services. Note that the T_{fn} should be much more restrictive than the T_{fp}, such that $T_{fn} \leq T_{fp}$. We cannot accept that anomalies within a CI and its industrial resources are not detected properly, since they could lead errors or faults into cascading [3]. One way to know the situation and reliability of the entire system, would be to (periodically or on-demand) generate a report with accumulative values of the counters ($count_{tp}$, $count_{fp}$, $count_{fn}$) through the *Reporter* sub-module. Finally, and for extending the functionality of the approach, a *Diagnosis Manager* is also used to check the lifetime of the CHs. As the Diagnosis Manager of Section 3.1, it will have to frequently check whether a specific CH_j stopped

sending messages during a significant time period by analyzing its sent frequency in the cache memory. If this occurs, the manager will have to diagnose its existence by sending a message based on DMAP objects. If the CH_j does not respond within a maximum time limit, the manager will have to warn of the situation using the type of event *event_CH_discardNode*. These diagnoses allow the system to manage isolated areas caused by malfunctions or denial of service attacks in CHs. Obviously, this action should be carried out for each network node, but this could mean a degradation of performance. For this reason, we supervise the lifetime of sensors using the counter $counter_{SensorBh}$, and thus we avoid a increasing in the communication overhead.

3.3 Other Major Points of Discussion

It is quite important to define a suitable value for σ_{error} and an appropriate buffer size for Δ_{Bff} (See Section 3.2) for prevention. The higher the margin of error and the smaller the buffer are, the greater the probability of obtaining a high false positive rate. Figure 5 (left hand side Figure) shows this aspect and its importance for critical contexts. The values are obtained from a simulation executed under the Java platform, where a critical scenario has been implemented which is composed of three clusters with two or three sensors each, and the control of the network is managed by three (virtual) available operators. Sequences of events (intentionally stressed) have been analyzed according to different sizes Δ_{Bff1} (5, 10, 15), Δ_{Bff2} with value 5, and different values of σ_{error} (0.0, 0.010, 0.020, 0.030 and 0.040). Given this, Δ_{Bff} then takes the following values 10, 15, 20 ($\Delta_{Bff} = \Delta_{Bff1} + \Delta_{Bff2}$). For the generation of such event sequences, we have assumed the following criteria. Each sensor node periodically produces events with values that can range between 0 and 5. Each production maintains a special correlation with events transmitted in the recent past, such as the frequency of a particular type of event and its priority. If a type of event with a specific priority is significantly repeated in a short time period, a new type of event with a higher priority is generated.

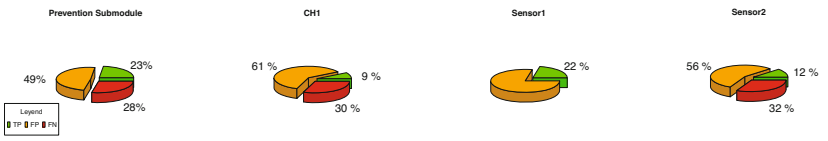


Fig. 6. A Report Obtained from a Simulation of an Critical Scenario (Intentionally Unstable)

As shown in Figure 5 (left hand side Figure), a system configured with a Δ_{Bff} size of 10 is less restrictive and precise than using a buffer with a size of 20. This is also the case when the system is configured with a σ_{error} with value of 0.040. On the other hand, Figure 5 (right hand side Figure) represents the importance of determining the sizes of Δ_{Bff1} and Δ_{Bff2}. The results indicate that a $\Delta_{Bff1} \geq \Delta_{Bff2}$ (continued line - $\Delta_{Bff1} = 10$ and $\Delta_{Bff2} = 5$; and $\Delta_{Bff1} = 10$ and $\Delta_{Bff2} = 10$) is more precise than using a $\Delta_{Bff1} < \Delta_{Bff2}$ (dashed line - $\Delta_{Bff1} = 5$ and $\Delta_{Bff2} = 10$). The reason is that the system

is able to contrast more present information with a small portion of past information so as to follow the behavior of the sensors in the time.

Although, all these configurations normally depend on the requirements of the SCADA organization and its security policies, they can change throughout of the life-cycle of the system. This change may occur when the Reporter Manager reports the current situation of the context. An example of a report could be the representation of percentages obtained from the values associated with the counters of TPs, FPs and FNs. Figure 6 shows said representation, which is also based on the results obtained from the simulation. In the extreme case that the counters of FPs and FNs are greater than their prescribed threshold (e.g., $count_{fp} > T_{fp}$), the SCADA Center could reconfigure the parameters to restrict the values associated to σ_{error} and Δ_{Bff}. On the other hand, it is essential to have good software maintenance of sensors, as their outputs are the input of the Prevention. This can be seen as a dependency relationship of 'cause-effect'. If a sensor does not work properly, the prediction can then tend to false positives or false negatives. Therefore, the role of the CH to detect malfunctions in sensors and the role of the Maintenance Manager to control anomalous behaviors in the entire system are fundamental to avoid disturbances in the final prediction.

4 Conclusions

A dynamic situational awareness model for control systems has been proposed here. The approach is based on the composition of different technologies and construction blocks in order to provide a set of benefits for situational awareness, such as *dissemination, prevention, detection, response, control, maintenance* and *safety-critical*. In particular, we have seen that we can obtain information from the infrastructure and its surroundings by using a WSN, and know their real states by managing different kinds of incidents. Through a hierarchical configuration, the system can detect particular malfunctions using simple behavior patterns, in addition to preventing and warning of the proximity of unstable situations, and responding to them in a timely manner. In addition, data redundancy enables the system to be aware of incidents that have occurred in the past, and recover the control when essential parts of the system remain isolated or out of service. Finally, it is worth highlighting that the design proposed in this paper can be extrapolated to other critical contexts such as transport systems.

Unfortunately, it is still necessary to continue further with the topic of situational awareness for protection of CIs to endow the system with autonomous and dynamic capacities. It would be interesting to explore new technologies and techniques and adapt them to the critical context without compromising its security and performance. Our next goal will be to extend the approach to consider all of these aspects, in addition to those topics related to security. In particular, this research will focus on open privacy issues to protect sensitive information within the cloud [5], and on designing simple behavior patterns to detect threats/attacks within a sensor network [11]. Note that parts of these topics are still very dependent on advances in hardware/software resources of sensor nodes. Therefore, investigation in this area is also needed.

Acknowledgments. This work has been partially supported by PISCIS (P10-TIC-06334), ARES (CSD2007-00004) and SPRINT (TIN2009-09237) projects. This last one is co-funded by the FEDER Program.

References

1. NIST Special Publication 1108R2, NIST Framework and Roadmap for Smart Grid Interoperability Standards, Release 2.0, Office of the National Coordinator for Smart Grid Interoperability (February 2012)
2. Falahati, B., Fu, Y.: A Study on Interdependencies of Cyber-Power Networks in Smart Grid Applications. In: 2012 IEEE PES Conference on Innovative Smart Grid Technologies, Washington DC, USA (January 2012)
3. Alcaraz, C., Lopez, J.: Analysis of Requirements for Critical Control Systems. In: Sixth IFIP WG 11.10 International Conference on Critical Infrastructure Protection. National Defense University, Washington, DC (2012)
4. ISA100.11a, ISA-100.11a-2009. Wireless systems for Industrial Automation: Process Control and Related Applications, The International Society of Automation (2009-2012)
5. Alcaraz, C., Agudo, I., Nunez, D., Lopez, J.: Managing Incidents in Smart Grids à la Cloud. In: IEEE CloudCom 2011, pp. 527–531. IEEE Computer Society (2011)
6. ENISA, Securing Europe's Information Society, Work Programme (2011)
7. Lopez, J., Roman, R., Alcaraz, C.: Analysis of Security Threats, Requirements, Technologies and Standards in Wireless Sensor Networks. In: Aldini, A., Barthe, G., Gorrieri, R. (eds.) FOSAD 2007/2008/2009. LNCS, vol. 5705, pp. 289–338. Springer, Heidelberg (2009)
8. Oxford Dictionary, Anomalous Situation, `http://oxforddictionaries.com/definition/anomalous` (retrieved on June 2012)
9. TinyOS Working Group, `http://www.tinyos.net/` (retrieved on June 2012)
10. Salfner, F.: Event-based Failure Prediction An Extended Hidden Markov Model Approach. PhD Thesis, Humboldt-Universität Berlin (2008)
11. Alcaraz, C., Lopez, J.: A Security Analysis for Wireless Sensor Mesh Networks in Highly Critical Systems. IEEE Transactions on In Systems, Man, and Cybernetics, Part C: Applications and Reviews 40(4), 419–428 (2010)

Identifying OS Kernel Objects for Run-Time Security Analysis

Amani S. Ibrahim, James Hamlyn-Harris, John Grundy, and Mohamed Almorsy

Centre for Computing and Engineering Software Systems
Faculty of ICT, Swinburne University of Technology
Melbourne, Australia
{aibrahim,jhamlynharris,jgrundy,malmorsy}@swin.edu.au

Abstract. As dynamic kernel runtime objects are a significant source of security and reliability problems in Operating Systems (OSes), having a complete and accurate understanding of kernel dynamic data layout in memory becomes crucial. In this paper, we address the problem of systemically uncovering all OS dynamic kernel runtime objects, without any prior knowledge of the OS kernel data layout in memory. We present a new hybrid approach to uncover kernel runtime objects with nearly complete coverage, high accuracy and robust results against generic pointer exploits. We have implemented a prototype of our approach and conducted an evaluation of its efficiency and effectiveness. To demonstrate our approach's potential, we have also developed three different proof-of-concept OS security tools using it.

Keywords: Operating Systems, Kernel Data Structures, Runtime Objects.

1 Introduction

An OS kernel has thousands of heterogeneous data structures that have direct and indirect relations between each other with no explicit integrity constraints, providing a large attack surface to hackers. In Windows and Linux Operating Systems (OSes), from our analysis nearly 40% of the inter-data structure relations are pointer-based relations (indirect relations), and 35% of these pointer-based relations are generic pointers (*e.g.* null pointers that do not have values, and void pointers that do not have associated type declarations in the source code). Such generic pointers get their values or type definitions only at runtime according to the different calling contexts in which they are used [1]. In such a complex data layout, the runtime memory layout of the data structures cannot be predicted during compilation time. This makes the kernel data a rich target for rootkits that exploit the points-to relations between data structure instances in order to hide or modify system runtime objects. Hence, accurately identifying the running instances of the OS kernel data structures and objects is an important task in many OS security solutions such as kernel data integrity checking [2], memory forensics [3], brute-force scanning [4], virtualization-aware security solutions [5], and anti-malware tools [6]. Although discovering runtime objects has been

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 72–85, 2012.

an aim of many OS security research efforts, these have many limitations. Most fall into two main categories: *Memory Mapping Techniques* and *Value Invariants Approaches.*

Memory mapping techniques identify kernel runtime objects by recursively traversing the kernel address space starting from the global variables and then follow pointer dereferencing until reaching running object instances, according to a predefined kernel data definition – for each kernel version – that reflects the kernel data layout in the memory [5,7,8]. However, such techniques are limited and not very accurate. They are vulnerable to a wide range of kernel rootkits that exploit the points-to relations between data structures instances to hide the runtime objects or point to somewhere else in the kernel address space. They require a predefined definition of the kernel data layout that accurately disambiguates indirect points-to relations between data structures, in order to enable accurate mapping of memory. However – to the best of our knowledge – all of the current efforts (with the exception of KOP [7] and SigGraph [4]) depend on the security expert's knowledge of the kernel data layout to manually resolve ambiguous points-to relations. Thus, these approaches only cover 28% of kernel data structures (as discussed by Carbone *et al.* [7]) that relate to well-known objects. They are also not effective when memory mapping and object reachability information is not available. Sometimes security experts need to make a high-level interpretation of a set of memory pages where the mapping information is not available *e.g.* system crash dumps. Incomplete subsets of memory pages cannot be traversed, and data that resides in the absent pages cannot be recovered. They have a high performance overhead because of poor spatial locality, as the problem with general-purpose OS allocators is that objects of the same type could scatter around in the memory address space. Thus traversal of the physical memory requires accessing several memory pages. Finally, they cannot follow generic pointer dereferencing as they only leverage type definitions, thus cannot know the target types of these untyped pointers.

Value-invariants approaches such as DeepScanner [9], DIMSUM [10] and SigGraph [4], use the value invariants of certain fields or of a whole data structure as a signature to scan the memory for matching running instances. However, such a signature may not always exist for a data structure [4]. Moreover, many kernel data structures cannot be covered by such value-invariant schemes. For example, it is difficult to generate value-invariants for data structures that are part of linked lists (single, doubly and triply), because the actual running contents of these structures depend on the calling contexts at runtime. In addition, such approaches do not fully exploit the rich generic pointers of data structures' fields, and are not able to uncover the points-to relations between the different data structures. The performance overhead of these approaches is extremely high, as they scan the whole kernel address space with large signatures, as they typically include most data structure fields in the signature.

Motivated by the limitations of these current approaches and the need to accurately identify runtime kernel objects from a robust view that cannot be tampered with, we have developed a new approach called DIGGER. DIGGER is capable of systematically uncovering all system runtime objects without any prior knowledge of the operating system kernel data layout in memory. Unlike previous approaches, DIGGER is

designed to address the challenges of indirect points-to relations between kernel data structures. DIGGER employs a hybrid approach that combines new value-invariant and memory mapping approaches, in order to get accurate results with nearly complete coverage. The value-invariant approach is used to discover the kernel objects with no need of memory mapping information, while the memory mapping approach is used to retrieve the object's details in depth including *points-to* relations (direct and indirect) with the other running data structures without any prior knowledge of the kernel data layout in memory. DIGGER first performs offline static *points-to* analysis on the kernel's source code to construct a type-graph that summarizes the different data types located in the kernel along with their connectivity patterns, and the candidate target types and values of generic pointers. This type-graph is used to enable systematic memory traversal of the object details. It is not used to discover running object instances. Second, DIGGER uses the four-byte pool memory tagging schema as a new value-invariant signature – that is not related the data structure layout – to uncover kernel runtime objects from the kernel address space.

DIGGER's approach has accurate results, low performance overhead, fast and nearly complete coverage, and zero rate of false alarms. We have implemented a prototype system of DIGGER and evaluated it on the Windows OS to prove its efficiency in discovering: *(i)* kernel runtime objects; *(ii)* terminated objects that still persist in the physical memory; and *(iii)* semantic data of interest in dead memory pages. To demonstrate the power of DIGGER, we also have developed and evaluated three OS security prototype tools based on it, namely, B-Force, CloudSec+ and D-Hide. B-Force is a brute force scanning tool. D-Hide is a tool that can systematically detect any hidden kernel object type not just limited to the well-known objects. CloudSec+, a virtual machine (VM) monitoring tool, is used in virtualization-aware security solutions to externally monitor and protect VM's kernel data.

Section 2 gives an overview on the kernel data problem and review of key related work. Section 3 presents our DIGGER approach and section 4 explores its implementation and evaluation. Finally we discuss results and draw key conclusions.

2 Background

In OSes we usually refer to a running instance of a data structure (or a data type) as an *object*. Locating dynamic kernel objects in memory is the most difficult step towards enabling implementing different OS security solutions, as discussed above. Efficient security solutions should not rely on the OS kernel memory or APIs to extract runtime objects, as they may be compromised and thus give false information. On the other hand, the complex data layout of an OS's kernel makes it challenging to uncover all system objects. Previous solutions limit themselves to the kernel static data *e.g.* system call and descriptor tables [11], or can reach only a fraction of the dynamic kernel data [2,12], resulting in security holes and limited protection.

It is challenging to check the integrity of kernel dynamic data due to its volatile nature. Dynamic data structures change during system runtime in location, values and number of running instances. Moreover, modifications to kernel dynamic data violate integrity constraints that in most cases cannot be extracted from OS source code.

This is because the data structure syntax is controlled by the OS code while their semantic meaning is controlled by runtime calling contexts. Thus, exploiting dynamic data structures will not make the OS treat the exploited structure as an invalid instance of a given type, or even detect hidden or malicious objects. For example, Windows and Linux keep track of runtime objects with the help of linked lists. A major problem with these lists is use of C null pointers. Modifications to null pointers violate intended integrity constraints that cannot be extracted from source code as they depend on calling contexts at runtime. This makes it easy to unlink an active object by manipulating pointers and thus the object becomes invisible for the kernel and for monitoring tools that depend on kernel APIs *e.g.* HookFinder [6] or memory traversal such as KOP [7], CloudSec [5], and OSck [8] – in addition to the limitations discussed above.

DeepScanner [9], DIMSUM [10], Gilbraltar *et al.* [2], and Petroni *et al.* [12] – as value-invariant approaches – are limited in that their authors depend on their knowledge with the kernel data layout, making their approach limited to a few structures. Also these tools do not consider the generic pointer relations between structures, making their approach imprecise and vulnerable a wide range of attacks that can exploit the generic pointers, in addition to high performance overhead due to large signatures.

To the best of our knowledge all existing approaches, whether value-invariant or memory traversal – with the exception of KOP [7], and SigGraph [4] – depend on the OS expert knowledge to provide kernel data layout definition that resolves the points-to relations between structures. SigGraph follows a systematic approach to define the kernel data layout, in order to perform brute force scanning using the value-invariant approach. However, it only resolves the direct points-to relations between data structures without the ability to solve generic pointers ambiguities, making their approach unable to generate complete and robust signatures for the kernel. KOP is the first and only tool that employs a systematic approach to solve the indirect points-to relations of the kernel data. However, KOP is limited in that: the points-to sets of the void * objects are not precise and thus they use a set of OS-specific constraints at runtime to find out the appropriate candidate for the objects. KOP assumes the ability to detect hidden objects based on the traditional memory traversal techniques which are vulnerable to object hiding. Moreover, both KOP and SigGraph have very high performance overhead to uncover kernel runtime objects in a memory snapshot.

Rhee *et al.* [13] propose an interesting approach to detect runtime objects by analysing object allocation and reallocation instructions executed. However their approach has quite high performance overhead and thus cannot be used in traditional OS security tools – only for advanced debugging tools. Also, despite the feature of detecting allocations and deallocations in near real time, they cannot even identify the object type. They need to analyse executed instructions offline to identify object type and details.

3 DIGGER Architecture

DIGGER's goal is to systematically uncover all kernel running objects in a memory snapshot or from a running VM without any prior knowledge of the kernel data layout. The high-level process of DIGGER is shown in Fig. 1. DIGGER has three main

components: *Static Analysis Component, Signature Extraction Component and Dynamic Memory Analysis Component,* discussed below in detail.

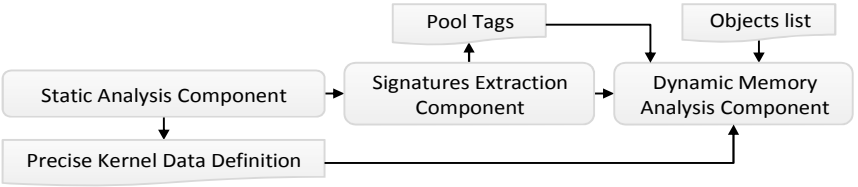


Fig. 1. The high-level process of DIGGER approach

3.1 Static Analysis Component

Performing static analysis on the kernel source code is the key to automate the process of extracting kernel objects' details without any prior knowledge of the kernel data layout. DIGGER first performs static *points-to* analysis on the kernel's source code to systematically solve the ambiguous *points-to* relations between kernel data structures by inferring the candidate target types and values for generic pointers. The result of the *points-to* analysis step is a kernel data definition represented as a type-graph. This type-graph precisely models data structures and reflects accurately both direct and indirect relations that reflect the memory layout of the data structures. The type-graph is **not** used to uncover kernel objects - it is used to retrieve running objects' detailed type structure. The level of details is selected by tool users based on the required hierarchy depth. This controls the trade-off between details and performance overhead, as some object types have hundreds of hierarchically-organised fields.

We build this type-graph using our tool KDD [14,15]. KDD performs interprocedural, context-sensitive, field-sensitive and inclusion-based *points-to* analysis on the kernel source code. KDD is able to perform highly precise and scalable *points-to* analysis for large C programs that contain millions lines of code *e.g.* OS kernel, without any prior knowledge of the OS structure.

3.2 Signature Extraction Component

It is difficult to obtain robust signatures for kernel data structures for the following reasons: **First**, data structure sizes are not small. From our analysis for Windows and Linux, we found that a single data structure could be several hundreds of bytes. Such big signatures increase the discovery cost and the performance overhead. **Second**, it is difficult to identify which fields of a target data structure can be used as scanning signatures to effectively detect stealthy malware and be difficult to be evaded. **Third**, the OS kernel contains thousands of data structures, making the process of generating "unique" signatures for this huge number of structures very challenging.

DIGGER makes use of the pool memory tagging schema of the kernel object manager to overcome the first two problems, and is motivated by the below paragraph from Windows internals book [16] (we call it WI-note) to overcome the third problem

– details discussed below: "*Not all data structures in the OS are objects. Only data that needs to be shared or made visible to user is placed in objects. Structures used by one component of the OS to implement internal functions are not objects*".

Windows kernels use pool memory to allocate the kernel objects. The pool memory can be thought of as a kernel-mode equivalent of the user-mode heap memory. When the object manager allocates a memory pool block, it associates the allocation with a pool tag – a pool tag is a unique four-byte tag for each object type. We use this tag as a value-invariant signature to uncover the kernel objects running instances. The pool tag list for the Windows OS can be extracted from the symbol information *e.g.* Microsoft Symbols. However, the pool tag is not enough to be an object signature. For instance, if we have a pool tag "Proc" and we scan the memory using the ASCII code of this pool tag, any word that has the same ASCII string will be detected as an object instance from that object type. Thus we need to add another checking signature that guarantees accurate results. We make use of the object dispatcher header (each allocated object starts with a dispatcher header that is used by the OS to provide synchronization access to resources). The first three bytes of the dispatcher header is unique for each object type, as they describe an object's type and size. These three bytes can be calculated from the generated type-graph (from our static analysis component). From our experiments, we found that those three bytes are static and cannot be changed during object runtime. Key features of using pool tags as signatures are: *(i)* not being tied to data structure layout and thus effective in different OS kernel versions where data structure layout change may occur; and *(ii)* the very small size of the signature that decreases performance overhead significantly.

To the best of our knowledge, all current OS security research for Windows and Linux treat all data structures as objects and do not consider the WI-note. This WI-note enables filtering the list of data structures extracted at the static analysis step, in order to obtain a list of the actual runtime object types. Each data structure that has a pool tag used by the Windows allocators is considered as an object and the other data structures are not. This massively reduces the number of object types from thousands to dozens. This solves the problem of generating unique signatures for such a huge kernel data structures size (the third obstacle), and also frees resources for analysis of the most important data structures. For the other data structures (non-objects that are less important than objects), we use the type-graph to uncover these data structures using the *points-to* relations of these data structures with the uncovered objects.

3.3 Dynamic Memory Analysis Component

The output of the memory analysis component is an object-graph whose nodes are instances of data structures and objects – in the memory snapshot – and edges are the relations between these objects. Using the pool tags and the additional checking signature, the dynamic memory component scans the kernel address space with eight byte granularity (default size of the pool header) to extract the runtime instances of the different kernel object types. However, until this step we can just identify that there is a running object instance of type *T* but we cannot know any details about the object itself or even the object name. When an object is being allocated by the object

manager it is prefixed by an object header and the whole object (including the object header) is prefixed with a pool header data structure, as shown in Fig. 2.

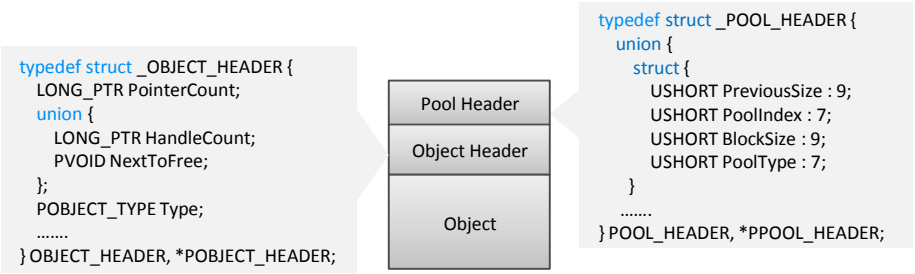


Fig. 2. The memory layout of allocated objects in the pool memory

The pool header is a data structure used by the Windows object manager to keep track of memory allocations. The most important fields in the pool header are the pool tag and the block size. These fields help in our algorithm to extract the object details as follows: **First**, Pool Tag; by subtracting the offset *f* of the pool tag field from the address *x* where an object has been detected (using the pool tag and the additional checking signature), we can get the pool block start memory address *y*. By adding the size of the pool header and the object header to *y*, we can calculate the object's start address. Then we retrieve the object's details based to our generated kernel type-graph – from the static component – by traversing the kernel memory. The size of the pool and object headers are calculated from the kernel type-graph. **Second**, Block Size; the block size field indicates the pool block size *s* that has been allocated for an object *O*. This field helps to speed up the scan process, by skipping *s* bytes of memory starting from the *y* address to reach the start address of next pool block or a kernel memory address.

We have two strategies for uncovering running kernel objects. **First**, for memory images; the size of a complete memory image is quite big and the kernel address space ranges from 1GB to 2GB in 32bit OSs and up to 8TB in 64bit OSs according to the memory layout used by the hardware and the available hardware memory. Scanning such a huge number of memory pages is too expensive. To solve this problem and get the fastest coverage for the kernel address space, we scan only the pool memory instead of the whole kernel address space. There are two distinct types of pool memory in Windows OS: paged pool and nonpaged pool. Both are used by the kernel address space to store the kernel and executive objects. The nonpaged pool consists of virtual memory addresses that are guaranteed to reside in physical memory as long as the corresponding kernel objects are allocated. The kernel uses the nonpaged pool memory to store the runtime objects that may be accessed when the system cannot handle page faults *e.g.* processes, threads and tokens. The paged pool consists of virtual memory that can be paged in and out of the system. This means that by scanning the nonpaged pool memory, which is a trusted source of information, we can get all the running objects instances that are potential target for hackers as they always reside

in physical memory. On uniprocessor or multiprocessor systems there exists only one nonpaged pool and this number can be confirmed using the global variable *nt!ExpNumberOfNonPagedPools*. The OS maintains a number of global variables that define the start and end addresses of the paged and nonpaged pool memory: *MmPagedPoolStart*, *MmPagedPoolEnd*, *MmNonPagedPoolStart* and *MmNonPagedPoolEnd*. These pointers can be used to speed up the scanning by limiting the scanned area. From our observations, we found pool memory size is around 4.5% and 8% from the kernel address space in 32-bit and 64-bit OS, respectively. **Second**, for un-mappable memory pages; in this case, the size of pages set is relatively small. We perform a scan on the whole set of the memory pages using the pool tag and the additional checking signature. However, as the memory mapping information may not be available in such un-mappable memory pages, not all of the details for the discovered objects can be retrieved as we depend on the memory traversal technique according to the generated type-graph.

4 Implementation and Evaluation

We have developed a prototype of DIGGER. The static analysis component was built using our previously developed tool, KDD [1,14]. The signatures and runtime components are standalone programs and all components are implemented in C#. The runtime component works: *(i)* offline on memory snapshot, raw dumps (*e.g.* dumps in the Memory Analysis Challenge and Windows crash dumps), and VMware suspended sessions. *(ii)* Online in a virtualized environment by scanning VMs' physical memory from the hypervisor level. We have evaluated the basic functionality of DIGGER with respect to the identification of kernel runtime objects and the performance overhead of uncovering these objects. We performed different experiments and implemented different OS security prototype tools to demonstrate DIGGER's efficiency. In section 4.1 we evaluate the static and runtime components, and their performance overhead. In section 4.2, we explore the implemented OS prototype tools, and finally in section 4.3 we discuss the main features and limitations of DIGGER.

4.1 Uncovering Objects

For the static analysis component, we applied KDD's static analysis to the source code of the Windows Research Kernel (WRK[1]) (a total of 3.5 million lines of code), and found 4747 type definitions, 1858 global variables, 1691 void pointers, 2345 null pointers, 1316 doubly linked list and 64 single linked lists. KDD took around 28 hours to complete the static analysis on a 2.5 GHz core i5 processor with 12 GB RAM. As our analysis was performed offline and just once on each kernel version, the performance overhead of analyzing kernels is acceptable and does not present any problem for any security application using KDD. The performance overhead of KDD could be decreased by increasing the hardware processing capabilities, as such types of analysis usually run with at least 32 GB RAM.

[1] WRK is the only available source code for Windows.

To enable efficient evaluation for the runtime component, we need a ground truth that specifies the exact object layout in kernel memory so that we can compare it with the results of DIGGER to measure false positive rates. We build the ground truth as follows: we extracted all data structure instances of the running Windows OS memory image *via* program instrumentation using the Windows Debugger (WD). We instrumented the kernel to log every pool allocation and deallocation, along with the address using the WD. In particular, we modified the GFlags (Global Flags Editor) to enable advanced debugging and troubleshooting features of the pool memory. We then measured DIGGER efficiency by the fraction of the total allocated objects for which DIGGER was able to identify the correct object type. We performed experiments on 3 different versions of the Windows OS on a 2.8 GHz CPU with 2GB RAM. Each memory snapshot size was 4GB. Table 1 shows the results of DIGGER and WD in discovering the allocated instances for specific object types in two of the three Windows versions (not showing all objects, for brevity).

Table 1. Experimental results of DIGGER and WD on Windows XP 32 bit and 64bit. Memory, paged and nonpaged columns reprsent the size in pages (0x1000 graunrality) of the kernel address space, paged pool and nonpaged pool, repectively. WD and DIG refer to WD's and DIGGER results. FN, FP and FP* denote the false negative, reported false positive and the actual false poitive rates, repectively.

Object	Windows XP 32bit					Windows XP 64bit				
	Memory		Paged		Nonpaged	Memory		Paged		Nonpaged
	915255		27493		11741	1830000		35093		17231
	WD	**DIG.**	**FN %**	**FP %**	**FP*%**	**WD**	**DIG.**	**FN %**	**FP %**	**FP*%**
Process	119	121	0.00	1.65	0.00	125	125	0.00	0.00	0.00
Thread	2032	2041	0.00	0.44	0.00	2120	2121	0.00	0.04	0.00
Driver	243	243	0.00	0.0	0.00	211	211	0.00	0.00	0.00
Mutant	1582	1582	0.00	0.0	0.00	1609	1609	0.00	0.00	0.00
Port	500	501	0.00	0.19	0.00	542	542	0.00	0.00	0.00

From table 1 we can see that DIGGER achieves zero false negative rates, and a low false positive rate. However, from our manual analysis of the results, we found that this reported false positive rate is not an actual false positive. This difference represents deallocated objects that still persist in the physical memory after termination; we call these "dead memory pages objects – DMAO". They are present because the Windows OS does not clear the contents of memory pages to avoid the overhead of writing zeroes to the physical memory. However, we noticed from our analysis that the pointer and handle count of the DMAO is always zero. This enables differentiating the active objects from the DMAO, and thus our actual false positive rate becomes zero (FP*). We argue this finding thus: whenever the kernel has to allocate a new object it will return the pool block address from the pool free list head. For example, the EPROCESS structure of a newly created process will overwrite the object data of a process that has been terminated previously. This because when a block is freed using the `free` function call, the allocator just adds the block to the list of free blocks

without overwriting memory. These DMAO can provide forensic information about an attacker's activity. Imagine an attacker runs stealthy malware and then terminates it on a victim machine. After the termination there may still exist for a non-trivial period of time some forensic data of interest in the dead memory pages. To prove our assumption, we analyzed the dead memory pages in order to uncover semantic data of interest for the some terminated processes. However, our approach can work for any other object type. We used some benchmark programs to run in three memory images and then analyzed the dead memory pages to uncover some data of interest: user login information (GroupWise email client), chat sessions (Yahoo messenger), FTP sessions (FileZilla). We created 9 processes (three of these are the benchmark programs) and performed some CPU-intensive operations using these processes. We terminated these processes after 5 hours, 2 hour and 15 minutes in three different memory images – identified L, M and S, respectively. Then we created 4 different (new) processes 5 minutes after termination. The memory images were then scanned for runtime objects using DIGGER's runtime component. We found that 3 from the terminated processes' physical addresses were overwritten by EPROCESS structure for new processes, while another three processes (from the terminated ones) still persisted in memory (at the same address in the memory). We made the following observations. First, for the email client we were not able to identify the login information (user name and password) for all of the memory images. For the ftp client we were able to identify the server name, and the server and client connection ports for the S image only, without any ability to locate the login credentials in all of the three images. For the chat benchmark application, we were able to locate the username, the connection port and some recent chat sessions in the S image only. This data recovery approach is not effective if the program zeros its memory pages before termination.

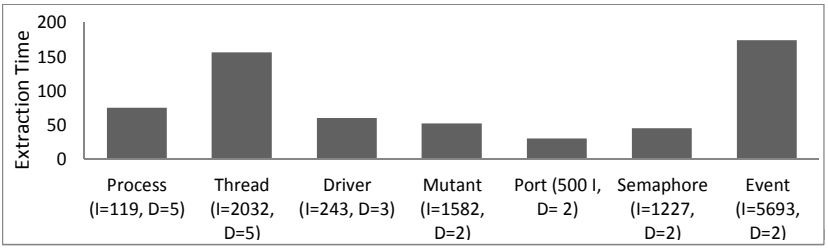


Fig. 3. Object details extraction normalized time

We have evaluated DIGGER's runtime performance to demonstrate that it can perform its memory analysis in a reasonable amount of time. We measured DIGGER running time when analyzing the memory snapshots used in our experiments. The median running time was around 0.8 minutes to uncover 12 different object types from the nonpaged pool, and 1.6 minutes to uncover another 15 object type from the paged pool. This time included the time of loading the memory snapshot from the disk to the runtime analysis component. We consider this running time to be acceptable for offline analysis and even for online analysis in virtualized environments. This is because DIGGER is able to detect the DMAO that could be created and terminated

between the scan time intervals. However, we cannot argue that it would be 100% accurate. Comparing DIGGER with SigGraph [4], DIMSUM [10], KOP [7], CloudSec [5]: DIGGER is the fastest with highest coverage and lowest performance overhead. The performance overhead of extracting object details based on our generated type-graph differs according the required details-depth. Fig. 3 shows the time consumed (in seconds) to extract object details with different depths for all of the running instances from a specific object type. "I" denotes the number of the running objects from the object, and "D" denotes the depth of the extracted details.

4.2 Security Applications

We have also evaluated DIGGER by developing three prototype OS security tools to demonstrate its efficiency. We chose these applications because they address common important OS security tools. Our experiments with these tools have demonstrated DIGGER's efficiency and the false alarms rate is similar to what discussed in table 1.

Object Hiding Detection. Previous efforts have focused on detecting specific types of hidden objects by hard-coding OS expert knowledge of the kernel data layout [12]. Other approaches rely on value-invariants such as the matching of process list with the thread scheduler [17]. Other approaches are based on logging malware memory accesses [18,19] and provide temporal information. However they can only cover known attacks and cannot properly handle zero-day threats. All of these approaches are time-consuming, and require a human expert with deep knowledge of the system to create the rules and thus cannot cover all system objects. There are some approaches such as Antfarm [20] that track the value of the CR3 register. Although this approach is useful in a live environment, it cannot be used for memory forensics applications and the performance overhead of such an approach is very high. Given DIGGER's ability to uncover kernel objects, we developed a tool called D-Hide that can systematically uncover all kinds of stealthy malware (not just limited to specific object type, as done to date), by detecting their presence in the physical memory. We used DIGGER's approach to uncover the runtime kernel objects and then perform "external" cross-view comparisons with the information retrieved from mapping the physical memory using our generated type-graph. In other words, the first view is DIGGER's view and the other view is the memory traversal view (we start from the OS global variables and then follow pointer dereferencing until we cover all memory objects). Discrepancies in this comparison reveal hidden kernel objects. We implemented a memory traversal add-on for the runtime component that takes our generated type-graph and based on that graph, we traverse the kernel address space. We evaluated D-Hide ability to identify hidden objects with four real-world kernel rootkit samples: FURootkit, FuToRootkit, AFX Rootkit and HideToolz. We used WinObj (a windows internal tool) to compare the results with D-hide. D-hide correctly identified all hidden objects with zero false alarms. D-Hide has two key advantages: *(i)* No need for deep knowledge of the kernel data layout, as it depends on DIGGER static component to get an accurate kernel data layout. *(ii)* D-Hide can perform cross-view comparison without the need for any internal tools *e.g.* task manager or WinObj that gets the internal view, as done in the current cross-view researches. This feature enables deploying D-hide in VMs hosted in the cloud platform

where the cloud providers do not have any control over VMs, as discussed in [11]. *(iii)* D-Hide is unlike previous tools [21,17] that rely on authors' knowledge of the kernel data and thus is not limited to specific objects.

Brute Force Scanning Tool. Given a range of memory addresses and a signature for a data structure or object, brute force scanning tools can decide if an instance of the corresponding data structure exists in the memory range or not [4]. Brute force scanning of kernel memory images is an important function in many operating system security and forensics applications, used to uncover semantic information of interest *e.g.* passwords, hidden processes and browsing history from raw memory. Given DIGGER's ability to uncover kernel objects, we developed B-Force – a brute force tool. From our experiments with five different small crash dumps of small sizes ranging from 12MB to 800MB, we found out that this method is highly effective with zero rates of alarms. However, this method could reveal false positives if the memory page set does not contain the pool header (that contains the pool tag) of the pool block along with the first three bytes of the object itself (to perform the additional signature checking). However, from our point of view this is unlikely, as the single memory page size is big enough to contain tens of pool blocks.

Virtual Machines Monitoring. We modified our earlier-developed VM monitoring tool, *CloudSec* [5], to use DIGGER to online analyze a Virtual Machine's (VM) memory of a running OS and extract all kernel running objects. *CloudSec* is a security appliance that monitors a VMs memory from outside the VM itself, without installing any security code inside the VM. *CloudSec* successfully uncovered and correctly identified the running kernel objects, with zero rate false alarms. The performance overhead of *CloudSec* to uncover the entire kernel running objects was around 1.1, 1.9 and 2.8 minutes with 0-level, 1-level and 2-level depths, respectively for a VM with a 2.8 GHz CPU and 4GB RAM. VM was executed under normal workload (50 processes, 912 threads, etc.). We can see that the performance overhead of scanning a VM's memory online is less that scanning a memory image, as access to VM's memory *via* hypervisors is faster than uploading a memory image to the analysis tool.

4.3 Discussion

DIGGER's approach provides a robust view of OS kernel objects not affected by the manipulation of actual kernel memory content. This enables development of different OS security applications as discussed in section 4.3, in addition to enabling systematic kernel data integrity checks based on the resultant object-graph. The key features of DIGGER include: **first**, the systematic approach it follows to extract OS kernel data layout and to disambiguate the points-to relations between data structures, all without any prior knowledge of the OS kernel data layout. **Second**, the robust and quite small signature size to uncover runtime objects, enhancing performance.

As the pool memory concept is related to Windows OSes, the current approach used in DIGGER's runtime component can only be used to analyze Windows OSes. DIGGER's runtime component is not related to a specific version of the Windows OS kernel and can work on either 32bit or 64 bit layout – SigGraph, DIMSUM and KOP

are also limited to a specific OS. However, the same approach can be used in Linux using the slab allocation concept. Slab allocation can be thought of as a pool memory equivalent of the Windows OS. Slab allocation is a memory management mechanism for Linux and UNIX OSes for allocating kernel runtime objects efficiently. The basic idea behind the slab allocator is having caches (similar to the pool blocks in Windows OS) of commonly used objects kept in an initialized state. The slab allocator caches the freed object so that the basic structure is preserved between uses to be used by a newly allocated object of the same type. The slab allocator consists of caches that are linked together on a doubly linked list called a *cache chain* that is similar to the list head of the pool memory used in Windows kernel. The static analysis component of DIGGER (KDD) can be applied on any C-based OS *e.g.* Linux, BSD and UNIX to perform highly detailed and accurate points-to analysis for the kernel data layout.

5 Summary

Current state-of-the-art tools are limited in accurately uncovering the running instances of kernel objects. We presented DIGGER, a new approach that enables uncovering dynamic kernel objects with nearly complete coverage and accurate results by leveraging a set of new techniques in both static and runtime components. Our evaluation of DIGGER has shown its effectiveness in uncovering system objects and in supporting the development of several OS security solutions.

Acknowledgement. The authors are grateful to Swinburne University of Technology and FRST Software Process and Product Improvement project for support for this research.

References

1. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J., Almorsy, M.: Supporting Virtualization-Aware Security Solutions using a Systematic Approach to Overcome the Semantic Gap. In: Proc. of 5th IEEE International Conference on Cloud Computing, Hawaii, USA (2012)
2. Baliga, A., Ganapathy, V., Iftode, L.: Automatic Inference and Enforcement of Kernel Data Structure Invariants. In: Proc. of 2008 Annual Computer Security Applications Conference, pp. 77–86 (2008)
3. Andreas, S.: Searching for processes and threads in Microsoft Windows memory dumps. Digital Investigation 3(1), 10–16 (2006)
4. Lin, Z., Rhee, J., Zhang, X.: SigGraph: Brute Force Scanning of Kernel Data Structure Instances Using Graph-based Signatures. In: Proc. of 18th Network and Distributed System Security Symposium, San Diego, CA (2011)
5. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J., Almorsy, M.: CloudSec: A Security Monitoring Appliance for Virtual Machines in the IaaS Cloud Model. In: Proc. of 2011 International Conference on Network and System Security (NSS 2011), Milan, Italy (2011)
6. Yin, H., Liang, Z., Song, D.: HookFinder: Identifying and understanding malware hooking behaviors. In: Network and Distributed Systems Security Symposium, NDSS (2008)

7. Carbone, M., Cui, W., Lu, L., Lee, W.: Mapping kernel objects to enable systematic integrity checking. In: Proc. of 16th ACM Conference on Computer and Communications Security, Chicago, USA, pp. 555–565 (2009)
8. Hofmann, O.S., Dunn, A.M., Kim, S.: Ensuring operating system kernel integrity with OSck. In: Proc. of 16th International Conference on Architectural Support for Programming Languages and Operating Systems, California, USA, pp. 279–290 (2011)
9. Liang, B., You, W., Shi, W., Liang, Z.: Detecting stealthy malware with inter-structure and imported signatures. In: Proc. of the 6th ACM Symposium on Information, Computer and Communications Security, Hong Kong, China, pp. 217–227 (2011)
10. Lin, Z., Rhee, J., Wu, C., Zhang, X., Xu, D.: Discovering Semantic Data of Interest from Un-mappable Memory with Confidence. In: Proc. of the 19th Network and Distributed System Security Symposium, San Diego, CA (2012)
11. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J.: Emerging Security Challenges of Cloud Virtual Infrastructure. In: Proc. of 2010 Asia Pacific Cloud Workshop Co-located with APSEC 2010, Sydney, Australia (2010)
12. Petroni, N.L., Fraser, T., Walters, A., Arbaugh, W.A.: An architecture for specification-based detection of semantic integrity violations in kernel dynamic data. In: Proc. of 15th Conference on USENIX Security Symposium, Canada, vol. 15 (2006)
13. Rhee, J., Riley, R., Xu, D., Jiang, X.: Kernel Malware Analysis with Un-tampered and Temporal Views of Dynamic Kernel Memory. In: Jha, S., Sommer, R., Kreibich, C. (eds.) RAID 2010. LNCS, vol. 6307, pp. 178–197. Springer, Heidelberg (2010)
14. Ibrahim, A.S., Grundy, J.C., Hamlyn-Harris, J., Almorsy, M.: Supporting Operating System Kernel Data Disambiguation using Points-to Analysis. In: Proc. of 27th IEEE/ACM International Conference on Automated Software Engineering, Essen, Germany (2012)
15. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J., Almorsy, M.: Operating System Kernel Data Disambiguation to Support Security Analysis". In: Proc. of 6th International Conference on Network and System Security (NSS 2012), Fujian, China. Springer (2012)
16. Russinovich, M., Solomon, D., Ionescu, A.: Windows Internals, 5th edn. Microsoft Press (2009)
17. Nanavati, M., Kothari, B.: Hidden Processes Detection using the PspCidTable. In: MIEL Labs (2010) (accessed November 2010)
18. Riley, R., Jiang, X., Xu, D.: Multi-aspect profiling of kernel rootkit behavior. In: Proc. of the 4th ACM European Conference on Computer Systems, Germany, pp. 47–60 (2009)
19. Xuan, C., Copeland, J., Beyah, R.: Toward Revealing Kernel Malware Behavior in Virtual Execution Environments. In: Kirda, E., Jha, S., Balzarotti, D. (eds.) RAID 2009. LNCS, vol. 5758, pp. 304–325. Springer, Heidelberg (2009)
20. Jones, S., Arpaci-Dusseau, A., Arpaci, R.: Antfarm: tracking processes in a virtual machine environment. In: Proc. of USENIX 2006 Annual Technical Conference, Boston (2006)
21. Jones, S.T., Arpaci-Dusseau, A.C., Arpaci-Dusseau, R.H.: VMM-based hidden process detection and identification using Lycosid. In: Proc. of the Fourth ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, USA, pp. 91–100 (2008)

Background Transfer Method for Ubiquitous Computing

Tae-Gyu Lee and Gi-Soo Chung

Korea Institute of Industrial Technology (KITECH), Ansan, Korea
tigerlee88@empal.com, gschung@kitech.re.kr

Abstract. Recently, as smart computing is constantly growing in terms of data and the number of users, the demand for memory and network resources on a wireless mobile terminal has increased rapidly. To accommodate the need for terminal and network resources, previous techniques have been studied for efficient use of the limited network channels and terminal memory resources. This paper presents a new transmission method which overcomes the data transmission limitations of wireless-handsets such as constraints on the transmission channels and storage capacity of a wireless smart terminal in ubiquitous computing. The wireless device's transmission and storage capacity limitations have hindered the advancement of ubiquitous computing on wireless Internet. This study proposes a real-time background transfer method to overcome these problems for powerful data transmission and large storage capacity among ubiquitous computing items.

Keywords: Transfer method, ubiquitous computing, mobile network, transfer recovery.

1 Introduction

Ubiquitous computing technology is located in the heart of ubiquitous computing as next generation computing method. It supports a user-oriented interface that greatly enhances the interaction between humans and computing devices through mobilization, miniaturization, and flexibility. Also, the ubiquitous computing technology makes it possible to make wearable, lightweight devices that can be embedded in the garments or fabrics. Additionally, in order to increase the user's mobility and accessibility to information, it aims to provide a mobile service platform that falls within the boundary between humans and information service media by building a ubiquitous external environment that is not restricted to time and space [1][2].

The ultimate goal of ubiquitous computing is to allow users to receive the best real-time information services wherever they are. The transmission and processing technology of the wireless information has occupied an important position.

Recently, the number of users and the utilization rate of wireless terminals continue to grow with the development of broadband wireless networks. By linking with the wireless internet services, the contents and categories of wireless mobile users become more varied. In particular, the present smart terminal services create a demand for new information, extend multimedia wire internet services to wireless network space, and constantly increase wireless mobile-oriented services.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 86–99, 2012.

However, despite the developments in wireless telecom services and mobile wireless resources such as wireless transmission channels, bandwidth, battery life, and memory are physically limited. These restrictions limit the reliability and real-time transfers of ubiquitous transmission systems [3][4][5].

In this paper, a new transmission method is proposed to overcome the transfer limitations from wireless terminal to wireless terminal communications; the limitations are caused by the constraints of the channel and memory resources of wireless handsets that also include smart phones. When you extend wireless internet service, the limitation of transmission and storage capability of the wireless terminal hinder the development of wireless multimedia services [6]. As an alternative to data transfer and storage issues of these mobile computing devices, a real-time background transfer system is proposed in this paper. The background support system improves the reliability and availability of the previous wireless transmission systems by supporting the real-time transfer technique through the real-time replication process of the transfer process and data, and by providing a transfer failure-recovery technique through monitoring and repairing the transmission failure between the clients who transfer the information.

This paper describes the following sequence. Section 2 describes a transfer method for a ubiquitous background transport system. Section 3 discusses experimentation and analyzes various performance evolutions of the proposed background transfer method. Finally, Section 4 describes the conclusions of this paper and future research directions.

Definition 1. *Background transfer* is an overlay (duplicate) transfer-storage process that is supported by a background system regardless of the smart terminal directly embedded into a mobile device. The background transfer is to provide stable and hard real-time transfer of all transport-related processes performed by a mobile user in the foreground.

2 Background Transfer Method

2.1 Background Transfer Process

The proposed background transmission system can be divided into background control process (*Control Process*), background data sending process (*Send Process*), and data receiving process (*Receive Process*) shown in Fig.1.

First, the background control process on a background server remains in a waiting status until it is called by a background-transfer request from a client (line 2). A sender C_i should submit the transfer request event, *request*_C_i, for a controller or a receiver C_j (line 16). When the background controller or the receiver receives the required message, *request*_C_i, each response *ACK/NAK* confirms message to sender C_i (lines 3-6, 29-33). If the sender C_i has received the *ACK* message, it transmits its data source and its data source identifier (URI_i) to the receiver C_j and the background controller (lines 17-20). The sender C_i selects a data source which he wants to transfer on his smart terminal or its storage space (CS_i) on a background server. The sender C_i and CS_i on the background server is a 1:1 mapping. (lines 19, 22, 25). Then, if the receiver

C_j and the controller completed the data receiving process, they reply *ACK* message to the sender C_i (lines 7-10, 34-37). Otherwise, they reply *NAK* message to the sender and the controller (lines 11, 38). Finally, if the controller receives *NAK* from C_j and if it has the same data on URI_i of itself, it retransmits data to the receiver C_i (lines 12-13). Otherwise, if the sender C_i receives *NAK* from both C_j and the controller, it retransmits data to the receiver C_i and the controller (lines 21-22). Otherwise, if the sender C_i receives *ACK* from the C_j and *NAK* from the controller as well as *Multi_FLAG* is *ON*, it just retransmits data to the controller (lines 23-26). The *Multi_FLAG* indicates whether the sender wants to multicast data to multi-receivers.

```
1.  Control Process on Controller(Multi_FLAG):
2.    initialize and wait;
3.    if(receive(request_Ci)== ok && ready ==ok) then {
4.      authenticate Ci;                    /* authentication of sender Ci */
5.      send ACK to Ci;                     /* confirm message */
6.    } else send NAK to Ci;
7.    if(receive(Data)== ok) then {
8.      receive(URIi and Data) from Ci;     /* receive data and URIi from sender Ci */
9.      save Data into URIi;                /* transmit data_copy to background server */
10.     send ACK to Ci;                     /* acknowledgement message */
11.   } else send NAK to Ci;
12.   if(reponse_Cj == NAK && response_Controller==ACK)
13.     send Data to Cj from URIi;          /* retransmit to receiver Cj */
14. End.

15. Send Process(Multi_FLAG):
16.   send request_Ci to Cj and Controller ;  /* request command */
17.   if(receive(response_Cj)==ok) {
18.     authenticate Ci to Cj and Controller;  /* authentication of sender */
19.     send Data and URIi to Cj and Controller;  /* transmit URIi and Data */
20.   };
21.   if(reponse_Cj == NAK && response_Controller==NAK) then
22.     send Data to Cj from URIi;          /* retransmit to receiver Cj */
23.   else if(reponse_Cj == ACK && response_Controller==NAK) then {
24.     if(Multi_FLAG ==ON)
25.       send Data to Controller form URIi;
26.   };
27. End.

28. Receive Process:
29.   if(receive(request_Ci)==ok && ready ==ok) {
30.     authenticate Ci or Controller;      /* authentication of sender or controller*/
31.     send ACK to Ci or Controller;       /* confirm message*/
32.   }
33.   else send NAK to Ci;
34.   if(receive(Data)==ok) {
35.     receive Data from Ci or Controller;  /* receive data form sender or controller */
36.     send ACK to Ci or Controller;       /* transfer completion message*/
37.   }
38.   else send NAK to Ci or Controller;
39. End.
```

Fig. 1. Background transfer processes

2.2 Data Transfer Model

The typical local data transmission model of Fig.2-(a)-(1) has both sender and receiver located within the same base station. This model has transfer characteristics;

the data transmission time is short, and the level of traffic congestion and transmission error rate is limited in a cell. In the existing global data transfer model of Fig.2-(a)-(2), a sender and a receiver are located in different base stations. This model has the transfer characteristics; the data transfer time is long and both traffic congestion levels and transmission error rates span across multiple cells. When efficient recovery services are required due to a network failure or congestion, the data copy will be retransmitted from the sender's terminal.

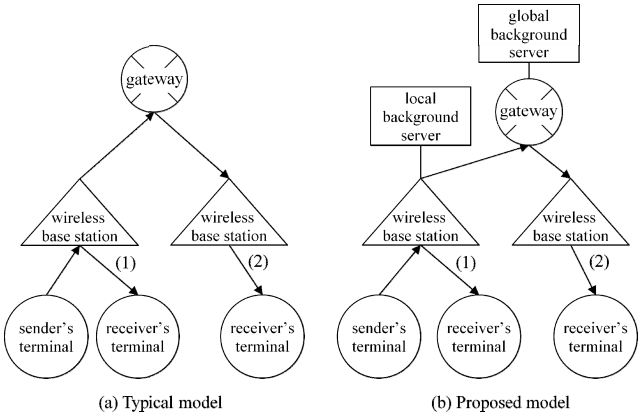


Fig. 2. Transfer system model

As shown in Fig.2-(b)-(1), when a sender transmits data to a receiver through the base station, the local background transfer model supports transmitting the data copy to the receiver as well as to the background server connected to the same base station at the same time. As shown in Fig.2-(b)-(2), when a sender transmits data to a receiver through the multiple base stations, the proposed global background transmission model simultaneously transmits the same data to the receiver as well as to the background server through a gateway connected to the base stations. If data packets are lost during transmission, or if the same receiver wants to download a data copy after the transfer is completed, the required data copy on the sender area of the background server (not on the sender terminal) will be retransmitted.

Local mobile networks build the background server on wireless base stations, while global mobile networks build the background server on gateway or Internet center.

3 System Performance Evaluation

3.1 Experimental Environment

The experimental specification of this study is based on the mobile networks such as the Bluetooth (802.15.3) wireless network for ubiquitous computing communications [12]. For background wired network construction, the IEEE Fast Ethernet standard is assumed for the lower bound of wired networks [13][14].

The proposed background transfer method will overcome the wireless resource constraints of the conventional method with bit rates lower than 100 Mbps because it takes advantage of wired transmission bit rates greater than 1 Gbps, which is supported by the transmission speed of the backbone network.

The simulation employs NS2 Simulator ver. 2.28 based on the configuration in Fig. 3. The transfer speed between the gateway and background server is up to 1 Gbps; the upload speed of the sending terminal is up to 1 Mbps; and the download speed of the receiving terminal is up to 1 Mbps. Under the same conditions, both the existing system and the proposed system are evaluated to analyze transmission performance. This experiment is classified as local and global transmission system based on consideration of the network path length from a sending terminal to a receiving terminal.

The purpose of this experiment is to analyze data transfer in the wireless environment. This experiment can be deployed to the general background transfer method by extending range of data transfer analysis to the wired terminal on the wired networks such as base station. In addition, it may increase the length (i.e. steps or hops) of the network from the gateway, and can be configured to interact with Internet. The position of deploying background servers based on failure analysis model can be optimized.

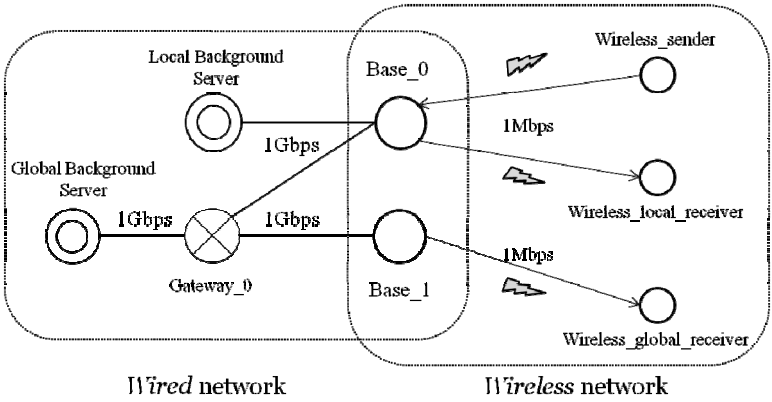


Fig. 3. Experimental network configurations

3.2 System Analysis

The transmission system analysis between the sending terminal and the receiving terminal are divided by the local data transfer analysis via a single base station, and the global data transmission analysis via two or more base stations.

Definition 2. *Transfer Networks* are classified into local and global data transfer networks. The *local networks* have both sender and receiver within one-hop base station. The other network in which that has two more hops is defined to *global networks*.

3.2.1 System Parameters

The principle transmission parameters of this study are the transmission capacity of the data source (V_s), the bandwidth of the transmission path (T_p), transmission time delay (D_p), download transmission error rate (E_r), upload transmission error rate (E_u), and so on. The following cost function (C_f) of Eq. 1 defines the performance evaluation for the existing transmission system and the proposed background transmission system based on these transmission parameters. The cost function (C_f) means the degree of transmission time which evaluates the throughput (or bandwidth), T_p, compared to the volume of the user's data source, V_s, including the condition of transmission failure rate.

$$C_f = \frac{V_s}{T_p} + \left(E_u + E_r\right) \cdot \left(\frac{V_s}{T_p}\right) \tag{1}$$

Definition of parameters in the cost function, C_f, of transmission time (expressed in seconds) is as follows. V_s is the total volume of source data transferred (measured in bytes). T_p can be applied to the throughput parameters of T_pU_w, T_pU_l, T_pU_g, T_pU_b, T_pD_b, T_pD_g, T_pD_l, and T_pD_w in bps according to transmission network structure. T_pU_w is the throughput on the transmission path from a wireless sending terminal to the nearest base station. T_pU_l is the throughput from the base station to the local background server. T_pU_g is the throughput from the base station to the gateway. T_pU_b is the throughput on the transmission path from the gateway to the global background server. T_pD_b is the throughput from the global background server to the gateway. T_pD_g is the throughput on the transmission path from the gateway to the base station. T_pD_l is the throughput from the local background server to the base station. T_pD_w is to the throughput from the base station to a wireless receiving terminal. E_r and E_u is the bit error rate (BER) for the transfer networks. The values of E_r and E_u are equal in order to apply the same error rate from Eq. 2 to Eq. 7.

3.2.2 Local Data Transfer Analysis

The terminal-to-terminal data transfer structure within the same cell has the wireless transfer path of wireless upload and wireless download through a single wireless base station. An analysis of the existing local transmission system is figured as the following cost function, $C_{f\_old\_local}$ found in Eq. 2.

$$\begin{aligned} C_{f\_old\_local} = &\left(\frac{V_s}{T_pU_w} + \frac{V_s}{T_pD_w}\right) + E_u\left(\frac{V_s}{T_pU_w}\right) \\ &+ E_r\left(\frac{V_s}{T_pU_w} + \frac{V_s}{T_pD_w}\right) \end{aligned} \tag{2}$$

The proposed local background transport system should perform the background process of uploading and downloading in addition to the sender's wireless uploads and the receiver's wireless downloads. An analysis for local transport of the background system is equated by the cost function, $C_{f\_new\_local}$ found in Eq. 3.

$$C_{f\_new\_local} = \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p D_w}\right) + E_u\left(\frac{V_s}{T_p U_w}\right) + E_r\left(\frac{V_s}{T_p U_l} + \frac{V_s}{T_p D_l} + \frac{V_s}{T_p D_w}\right) \tag{3}$$

In the cost function, $C_{f\_new\_local}$, of the proposed local background system, the V_s/T_pU_l can be removed from the time delay to retransmit packets to recover packet errors because the transport cost, V_s/T_pU_l, using a transmission bandwidth of T_pU_l overlaps with the transport costs, V_s/T_pD_w of T_pD_w in time. Thus, the cost function of the proposed method is described in Eq. 4.

$$C_{f\_new\_local} = \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p D_w}\right) + E_u\left(\frac{V_s}{T_p U_w}\right) + E_r\left(\frac{V_s}{T_p D_l} + \frac{V_s}{T_p D_w}\right) \tag{4}$$

Fig.4 is a graph based on comparing the proposed background transmission system and the existing transport system which both $C_{f\_new\_local}$ and $C_{f\_old\_local}$ represent, respectively. In the case of specific conditions with transmission error rate of 0.1 and 0.3, try to evaluate the transmission time delay. In Fig.4, the *old_error_rate* refers error rates applied to the typical method of Fig.2-(a) and *new_error_rate* refers error rates applied to the proposed method of Fig.2-(b). The graph with the highest transmission delays represents a case of *old_error_rate_0.3*, and the graph with the lowest transmission delays represents a case of *new_error_rate_0.3*.

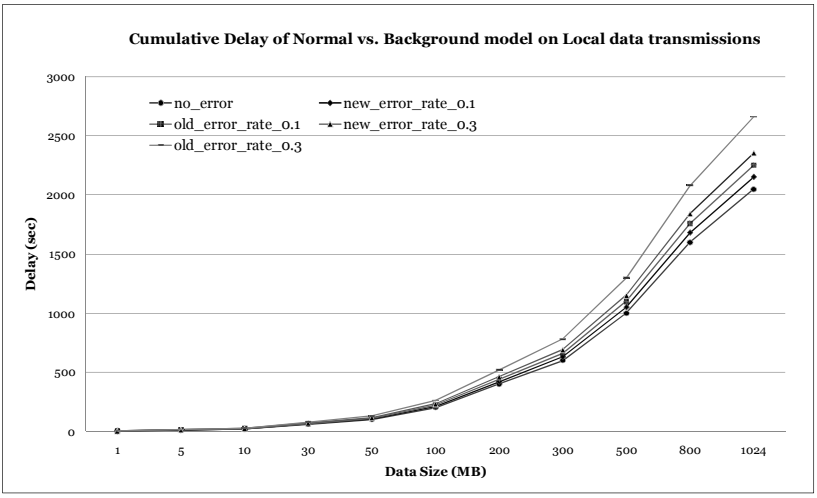


Fig. 4. Transfer delay on local transfer models

To evaluate the cost difference between the existing method and the proposed method, the difference of Eq. 4 from Eq. 2 is described as follows.

$$C_{f\_old\_local} - C_{f\_new\_local} = E_r V_s \left(\frac{1}{T_p U_w} - \frac{1}{T_p D_b} \right).$$

Therefore, $C_{f\_old\_local}$ is not necessarily greater than $C_{f\_new\_local}$. It depends on the value of $T_p U_w$ and $T_p D_b$.

When error rates increase, the performance difference between the proposed system and the existing system becomes greater (e.g. error rates in 0.1 and 0.3). In the proposed method, the *new_error_rate_0.3*, the graph of error rate 0.3 represents a shorter transmission time delay than the *new_error_rate_0.1*.

From the Eq. 4, when compared to the results of applying each transmission error rate of 0.1 and 0.3 respectively,

$$\begin{aligned} & C_{f\_new\_local}(E_u, E_r = 0.1) - C_{f\_new\_local}(E_u, E_r = 0.3) \\ &= 0.1 \times \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p D_l} + \frac{V_s}{T_p D_w} \right) - 0.3 \times \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p D_l} + \frac{V_s}{T_p D_w} \right) \\ &= -0.2 \times V_s \times \left(\frac{1}{T_p U_w} + \frac{1}{T_p D_l} + \frac{1}{T_p D_w} \right). \end{aligned}$$

This means that the higher failure rate takes a lot of transport costs. This proves that the retransmission from the background server is more effective than that from a wireless sender terminal in the wireless mobile environment that indicates higher transfer failure rate. Also, when the data transmission capacity increases, the performance difference of the proposed system and the existing system offers an even greater advantage (e.g. the transfer delay results in 1MB vs. 1024MB).

Therefore, in the local transport system, the greater the failure rate or the transfer volume for data is increased, the more the transmission time of the proposed system decreases compared to that of the existing system.

However, when data transmission capacity is low, for instance when the capacity is less than 100MB in Fig.4, the performance improvement is limited because the difference of cost between the proposed and the existed methods is not significant.

3.2.3 Global Data Transfer Analysis

The data transfer structure between sender terminal and receiver terminal located at different cells in typical global wireless transmission systems has transmission paths such as: wireless upload from a wireless terminal to a base station, wired upload from the base station to a gateway, wired download from the gateway to another base station, and wireless download from the base station to a wireless terminal. The existing global transmission system has the cost function, $C_{f\_old\_global}$, of Eq. 5.

$$C_{f\_old\_global} = \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p U_g} + \frac{V_s}{T_p D_g} + \frac{V_s}{T_p D_w}\right) + E_u\left(\frac{V_s}{T_p U_w}\right) + E_r\left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p U_g} + \frac{V_s}{T_p D_g} + \frac{V_s}{T_p D_w}\right) \tag{5}$$

The global background transmission system transmits data through the same transmission path as conventional wireless transmission system. In addition, it uploads the same data replication to the background server linked to a gateway and downloads a copy of the data being sent from the background server upon packet failure or data retransmission. The proposed system is evaluated in the cost function, $C_{f\_new\_global}$ (Eq. 6).

$$C_{f\_new\_global} = \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p U_g} + \frac{V_s}{T_p D_g} + \frac{V_s}{T_p D_w}\right) + E_u\left(\frac{V_s}{T_p U_w}\right) + E_r\left(\frac{V_s}{T_p U_b} + \frac{V_s}{T_p D_b} + \frac{V_s}{T_p D_g} + \frac{V_s}{T_p D_w}\right) \tag{6}$$

In the cost function, $C_{f\_new\_global}$, of the proposed global background system, the V_s/T_pU_b can be removed from the time delay to retransmit packets for recovering packet errors because the transfer cost, V_s/T_pU_b, using a transmission bandwidth of T_pU_b overlaps with the sum of the transmission costs, V_s/T_pD_g and V_s/T_pD_w, using the transmission bandwidth of T_pD_g and T_pD_w. Thus, the cost function of the proposed model is redeployed as the following in Eq. 7.

$$C_{f\_new\_global} = \left(\frac{V_s}{T_p U_w} + \frac{V_s}{T_p U_g} + \frac{V_s}{T_p D_g} + \frac{V_s}{T_p D_w}\right) + E_u\left(\frac{V_s}{T_p U_w}\right) + E_r\left(\frac{V_s}{T_p D_b} + \frac{V_s}{T_p D_g} + \frac{V_s}{T_p D_w}\right) \tag{7}$$

The graph that compares the proposed global background transmission system and the existing global transport system, which both $C_{f\_old\_global}$ and $C_{f\_new\_global}$ represent, respectively, is similar to the one in Fig. 4. In the case of specific conditions with transmission error rate of 0.1 and 0.3, the following sequence evaluates the transmission time delay. The graph with the highest transmission delays also represents a case of *old_error_rate_0.3*, and the graph with the lowest transmission delays represents a case of *new_error_rate_0.3*.

The difference between Eq. 7 and Eq. 5 which described as follows shows the cost difference between the existing and the proposed method.

$$C_{f\_old\_global} - C_{f\_new\_global} = E_r V_s\left(\frac{1}{T_p U_w} + \frac{1}{T_p U_g} - \frac{1}{T_p D_b}\right).$$

Therefore, $C_{f\_old\_global}$ is not necessarily greater than $C_{f\_new\_global}$. It depends on the value of T_pU_w and T_pD_b.

When error rates increase, the performance difference between the proposed system and the existing system become greater (e.g. error rates in 0.1 and 0.3). In the proposed method, the *new_error_rate_0.3* graph of error rate 0.3 represents a shorter transmission time delay than the *new_error_rate_0.1*.

Eq. 7 shows the results when applying each transmission error rate of 0.1 and 0.3 respectively,

$$C_{f\_new\_global}(E_u, E_r = 0.1) - C_{f\_new\_global}(E_u, E_r = 0.3)$$
$$= -0.2 \times V_s \times \left(\frac{1}{T_pU_w} + \frac{1}{T_pD_b} + \frac{1}{T_pD_g} + \frac{1}{T_pD_w} \right).$$

This means that the higher failure rate takes a lot of transport costs. This proves that the retransmission from the background server is more effective than that from the wireless sender terminal in the wireless mobile environment that indicates higher transfer failure rate. When the transmission capacity of data increases, the proposed system shows even greater improvements (e.g. the transfer delay results in 1MB vs. 1024MB).

Thus, in the global transfer system, the greater the failure rate or the volume data transmitted, the more the transmission time of the proposed system decreases compared to that of the existing system.

3.3 Performance Evaluations

This study makes a distinction between short-distance local transmission systems and long-distance global transmission systems, and it performs simulations and evaluates performance for each of the proposed systems and the existing system.

3.3.1 Local Performance Evaluation

In order to evaluate the transfer performance between a sending terminal and a receiving terminal existing within the same cell, the existing transfer scheme based on the client and the proposed scheme based on the background server are compared by applying each error rate from 0.1 to 0.3. The transmission error rate is the bit error rate (BER) for the transfer networks.

As shown in Fig. 5, for each error rate 0.1, 0.2, and 0.3, the transmission capacity (in bytes) of both the sender and the receiver were compared depending on the variance of the transfer time (from 0 to 150seconds) as the performance comparisons between the existing transmission scheme of terminal-to-terminal (i.e. *normal-rcv-put0.1*, *normal-rcv-put0.2*, and *normal-rcv-put0.3*) and the proposed background transmission scheme of server-to-terminal (i.e. *local-rcv-put0.1(bg)*, *local-rcv-put0.2(bg)*, and *local-rcv-put0.3(bg)*).

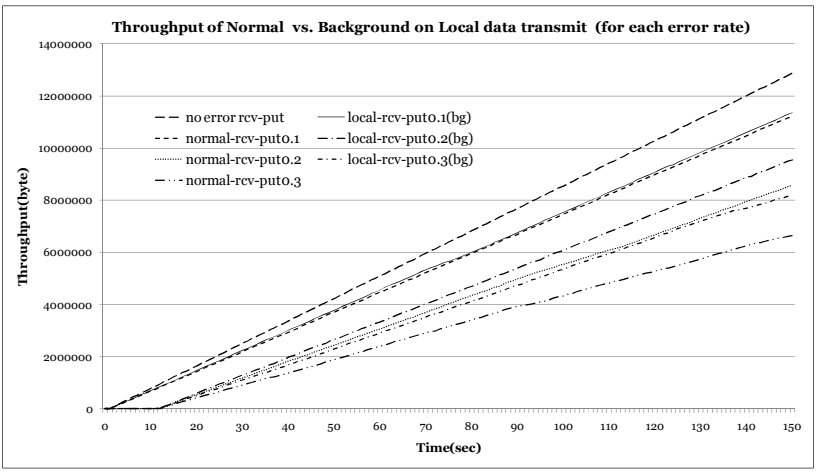


Fig. 5. Transfer throughputs on local transfer models

The top "no-error-rcv-put" graph in Fig.5 shows the case in which there are no errors. These results show that the total data transfer capacity of the proposed scheme (i.e. three graphs of the 2nd *local-rcv-put0.1(bg)*, the 4th *local-rcv-put0.2(bg)*, and the 6th *local-rcv-put0.3(bg)* in Fig.5), by supporting that the transmission storage on the background server network, is higher than that of the existing scheme (i.e. three graphs of the 3rd *normal-rcv-put0.1*, the 5th *local-rcv-put0.2*, and the 7th *local-rcv-put0.3* in Fig.5) based on the transport capacity of a wireless terminal with poor resources.

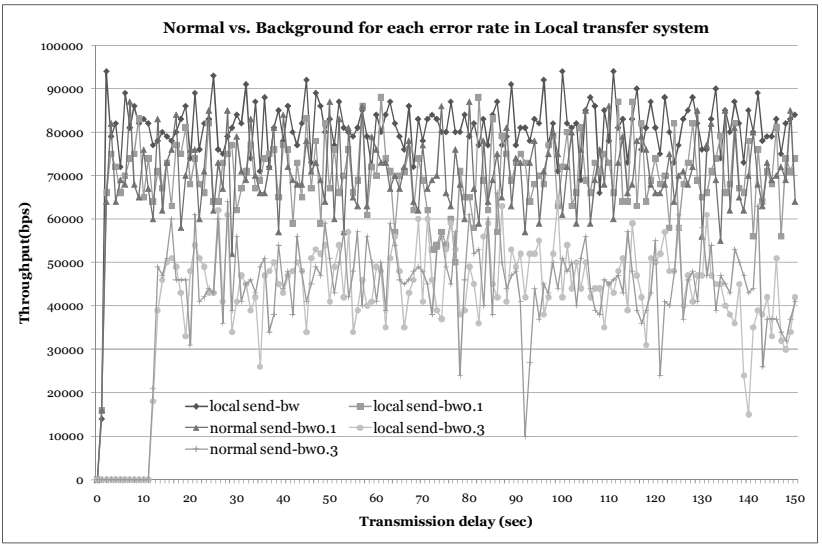


Fig. 6. Transfer throughputs by error rates on local transfer models

Then, in spite of transfer errors occurring in the local data transfer model, the proposed background transmission model supported recovery transmission for reliable data transmission. Fig.6 compares the transfer results based on a few transmission errors (e.g. 0.1 and 0.3) and that of an errorless case (i.e. 0.0) for 0 to 150 seconds. Fig.6 displays each transfer bit rate between the sender and the receiver terminal applied for the existing technique and the background transmission technique, respectively, to transmit the same data source in a wireless terminal.

The system overhead according to the gateway multiplexing is negligible when compared to the existing system. Therefore, the proposed background technique can be effectively supported without affecting the overhead of the existing gateway equipment.

3.3.2 Global Performance Evaluation

In order to evaluate the transfer performance between a sending terminal and a receiving terminal that exists within the different cells, the existing transfer scheme based on the client source and the proposed scheme based on the background server were compared by applying each error rate from 0.1 to 0.3. Transfer simulations were performed for both the existing scheme and the proposed background scheme based on the global transmission model. When the transmission capacity (in bytes) of the sender and the receiver is compared based on the time (varying from 0 to 150 seconds) as the performance comparisons of the existing transmission scheme of terminal-to-terminal and the proposed background transmission scheme of server-to-terminal, as the graphs shown in Fig.7, the results show that the proposed scheme, because of the supporting transfer capacity of the background server network, supports higher throughput than the existing method based on the storage capacity of the wireless terminals with vulnerable.

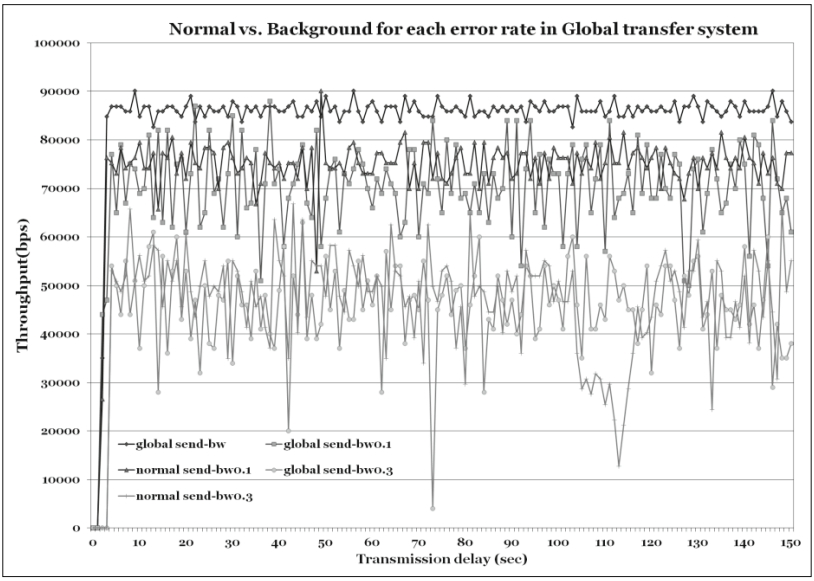


Fig. 7. Transfer throughputs by error rates on global transfer models

Then, in spite of transfer errors in the global data transfer model, the proposed background transmission model supported recovery transmission for reliable data transmission. Fig.7 compares the transfer results based on a few transmission errors (e.g. 0.1 and 0.3) and that of an errorless case (i.e. 0.0) for 0 to 150 seconds.

It appears that the system overhead added by the gateway multiplexing is negligible when compared to the existing system. Therefore, it is effectively feasible to support the background technique proposed in this study without replacing the existing gateway devices. The proposed background transport model supports the lightweight mobile users for smart computing applications in ubiquitous computing.

Because the data transfer of the sender and the receiving command of the receiver are being done in real time without any interference or intervention from the wireless network infrastructure, a data transfer process can strengthen the transmission reliability and real-time characteristics based on the backbone network.

4 Conclusion

From the analysis in Section 3, we can see that this study takes full advantage of the wired backbone network and the strengths of the background server in all aspects of data storage stability, transmission speed, and reliability.

While minimizing the intervention of wireless resources in the proposed background data transfer model, notice commands should be given to alert mobile clients about the choices of any transmission method.

Analysis of Fig.4 in Section 3.2 shows that the greater the failure rate or the data transport volume, the lower the average transmission time of the proposed background model compared to that of the existing model. However, if data transmission capacity is low (for example, less than 100MB in Fig.4), the performance improvement is limited because the cost difference of the proposed and the existed methods shows slightly. As the recent trend demonstrates, the proposed method is importantly recognized because the needs of data transmission users are continuously moved toward large amounts of data such as videos, images, and etc. Fig.5 of Section 3.3 shows that the data transmission capacity of the proposed model is approximately 30% greater than that of the existing model in the transfer process for 150seconds. Both Fig.6 and Fig.7 show that the transfer performance of the proposed model is constantly improved by 10% with the existing model.

References

1. McCann, J., Bryson, D.: Smart clothes and wearable technology, pp. 3–24, 205–213. CRC Press (2009)
2. Krumm, J.: Ubiquitous Computing Fundamentals, pp. 1–35. CRC Press (2010)
3. Xiao, Y., Rayi, V.K., Sun, B., Du, X., Hu, F.: A survey of key management schemes in wireless sensor networks. Computer Communications 30, 2314–2341 (2007)

4. Al-bar, A., Wakeman, I.: A Survey of Adaptive Applications in Mobile Computing. In: The 21st International Conference on Distributed Computing Systems Workshops (ICDCSW 2001), p. 246 (2001)
5. Ramana, K.S., Chari, A.A., Kasiviswanth, N.: A Survey on Trust Management for Mobile Ad Hoc Networks. International Journal of Network Security & Its Applications (IJNSA) 2(2), 75–85 (2010)
6. Goth, G.: Mobile Devices Present Integration Challenges. In: IEEE IT Pro., pp. 11–15 (May 1999)
7. Poslad, S.: Ubiquitous Computing-smart devices, environments and interactions, pp. 343–378. John Wiley & Sons (2009)
8. Xu, Y., Li, W.J., Lee, K.K.: Intelligent Wearable Interfaces, pp. 5–30. John Wiley & Sons Press (2008)
9. Dang, P.P., Chau, P.M.: Robust image transmission over CDMA channels. IEEE Transactions on Consumer Electronics 46(3), 664–672 (2000)
10. Chu, G.H.: Image transmission apparatus and method using CDMA communication network, U.S. Patent No. US007505782B2, March 17 (2009)
11. Siewiorek, D., Smailagic, A., Starner, T.: Application Design for Wearable Computing, pp. 51–58. Morgan & Claypool Publishers (2008)
12. Lo, A., Lu, W., Jacobsson, M., Prasad, V., Niemegeers, I.: Personal Networks: An Overlay Network of Wireless Personal Area Networks and 3G Networks. In: Third Annual International Conference on Mobile and Ubiquitous Systems: Networking & Services (July 2006)
13. Frazier, H.: The 802.3z Gigabit Ethernet Standard. IEEE Network 12(3), 6–7 (1998)
14. Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method and Physical Layer specifications," IEEE Standard for Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific requirements, Std. 802.3 (2008)

Selective Opening Chosen Ciphertext Security Directly from the DDH Assumption

Shengli Liu[1,3,⋆], Fangguo Zhang[2,3], and Kefei Chen[1,4]

[1] Dept. of Computer Science and Engineering,
Shanghai Jiao Tong University, Shanghai 200240, China
{slliu,kfchen}@sjtu.edu.cn
[2] School of Information Science and Technology
Sun Yat-sen University, Guangzhou 510006, China
isszhfg@mail.sysu.edu.cn
[3] State Key Laboratory of Information Security,
Institute of Software, Chinese Academy of Sciences
[4] Shanghai Key Laboratory of Scalable Computing and Systems, Shanghai

Abstract. Chosen-ciphertext security has been well-accepted as a standard security notion for public key encryption. But in a multi-user surrounding, it may not be sufficient, since the adversary may corrupt some users to get the random coins as well as the plaintexts used to generate ciphertexts. The attack is named "selective opening attack". We study how to achieve full-fledged chosen-ciphertext security in selective opening setting directly from the DDH assumption. Our construction is free of chameleon hashing, since tags are created for encryptions in a flexible way to serve the security proof.

Keywords: Selective opening security, Chosen-ciphertext attack.

1 Introduction

Indistinguishable Security against chosen-plaintext attack (IND-CPA) and security against chosen-ciphertext attack (IND-CCA2) have been considered as the security standard for public-key encryptions. However, IND-CPA and IND-CCA2 are not sufficient for some circumstances. For instance, consider an adaptive adversary in a secure multiparty computation setting, where the private channels connected between parties are implemented with public-key encryption. Now each of n senders sends a ciphertext $c_i = Enc(pk, m_i, r_i)$ to a receiver. The adaptive adversary may not only eavesdrop all the ciphertexts c_i, $i = 1, 2, \cdots, n$, over the channels, but also implement a so-called "selective opening attack" by corrupting a subset $I = \{i_1, i_2, \cdots, i_t\}$ of players ($t \leq n/2$). Each corrupted

⋆ Funded by NSFC (No. 61170229, 61070168, 60970111 and 61133014), Innovation Project (No.12ZZ021) of Shanghai Municipal Education Commission, and Specialized Research Fund (No.20110073110016) for the Doctoral Program of Higher Education.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 100–112, 2012.

player P_{i_l}, $l = 1, 2, \cdots, t$, will provide the adversary both the corresponding plaintext m_{i_l} as well as the random coin r_{i_l} that are used to generate the ciphertext c_{i_l}. The question is how to ensure the security of uncorrupted encryption c_i, $i \in \{1, 2, \cdots, n\} \setminus I$.

If the plaintexts m_i, $i = 1, 2, \cdots, n$, are independent of each other, then IND-CPA (IND-CCA2) is able to provide security against selective opening attack. However, multi-party computation always implies correlated plaintexts of the senders. As for correlated plaintexts m_i, Bellare, Dowsley, Waters and Scott Yilek proved that standard security does not imply security against selective-opening [15]. Therefore, it is necessary to define a new security notion, e.g. "encryption security against selective opening attack" [3].

Public-key encryption schemes with security against selective opening attacks (SOA) maintain security even in case of sender corruptions. If the adversary is given n ciphertexts and decides to open some of them, say $n/2$, the adversary will obtain both the plaintexts and the random encryption coins for the opened ciphertexts. "encryption security against selective opening attack" requires the security of remaining ciphertexts should be preserved.

1.1 Related Work

Security against Selective Opening Attack (SOA) has been recognized for 20 years [1], but the first schemes achieving SOA security were proposed in 2009 by the breakthrough work of Hofheinz, Bellare and Yilek [3,2,4]. In [3], a simulation-based semantic-style security for encryption under selective opening (SEM-SO-ENC) was formalized, while an indistinguishability-based version (IND-SO-ENC) was proposed to distinguish from the simulated-based one.

In [3], Bellare, Hofheinz and Yilek pointed out that any lossy encryption is IND-SO-ENC secure, and it is also SEM-SO-ENC secure if an additional property named "efficient openability" is satisfied. They also proposed an efficient construction of lossy encryption based on the DDH assumption. Lossy trapdoor functions, proposed by Peikert and Waters in 2008 [24], is a useful tool to achieve lossy encryption. As suggested in [25], sufficiently lossy trapdoor functions implies lossy encryption. Lossy trapdoor functions can be constructed based on the Decisional Diffie-Hellamn (DDH), Paillier Decisional Composite Residue (DCR), Decisional Quadratic Residue (DQR), and lattices-related assumptions [24,11,17,18,6,25,20]. That means IND-SO-ENC secure encryption schemes can be constructed based on the aforementioned number-theoretic assumptions. In 2012, Böhl et.al. gave a more rigorous definition about selective opening security.

In [5], Hemenway et.al. showed that both re-randomizable public-key encryption and statistically-hiding $\binom{2}{1}$-oblivious transfer imply lossy encryption. Combined with Hofheinz, Bellare and Yilek's result that lossy encryption is IND-SO-ENC secure, they got more efficient construction of IND-SO-ENC secure encryptions and SEM-SO-ENC secure ones.

The indistinguishability-based security definition (IND-SO-ENC) captures that the adversary cannot distinguish the encryption of actual messages from

that of random messages. In 2011, Hemenway et.al. [5] pointed out that IND-SO-ENC/SEM-SO-ENC only indicates security against chosen plaintext attacks, and IND-SO-ENC/SEM-SO-ENC is renamed by IND-SO-CPA/SEM-SO-CPA. They further generalized IND-SO-CPA to selective-opening security against chosen ciphertext attacks (IND-SO-CCA2/SEM-SO-CCA2). Chosen-ciphertext security (IND-SO-CCA2) in the selective opening setting captures that the adversary cannot distinguish the encryption of actual messages from that of random messages, even provided with a decryption oracle besides a corruption oracle. As shown in [5], CCA2 security is much more complicated in the selective opening setting, since the adversary is able to obtain the random coins of corrupted parties. Nevertheless, a paradigm was proposed to achieve IND-SO-CCA2 by Hemenway [5], who proved that a separable tag-based encryption scheme with selective-tag weak security in the selective-opening setting (i.e., IND-stag-wCCA2) could be transformed into a full-fledged IND-SO-CCA2 one with help of chameleon hashing.

Hemenway et al.[5] gave modular construction of IND-SO-CCA2 secure encryption from a certain family of tag-based encryption system using a variant of the Canetti-Halevi-Katz paradigm. However, since one-time signatures cannot be used (because the adversary would have to obtain one-time private keys for opened ciphertexts), they used chameleon hash functions to apply the CHK paradigm. The length of ciphertext is O(1), and the length of public/private key and computation complexity is of $O(n)$ with n the number of challenge ciphertexts. Hofheinz [19] showed how to avoid the $O(n)$ complexity for the sender and in the public/private key size. The construction also involves chameleon hash functions, and the security reduction is loose.

Fehr etal.[14] considered how to construct schemes with SEM-SO-CPA/SEM-SO-CCA2 security based on hash proof systems and L-Cross-Authentication codes. Their constructions are free of chameleon hash functions, but the number of group elements in the ciphertext is the same as the number of bits in the plaintext.

1.2 Our Contributions

In this paper, we focus on chosen-ciphertext security in the selective opening setting (IND-SO-CCA2). We show that a public-key encryption scheme with IND-SO-CCA2 security can be constructed directly based on the DDH assumption. Chameleon hashing is not necessary to achieve IND-SO-CCA2 security, since the IND-SO-stag-wCCA2 PKE scheme in [5] based on DDH can be modified and proved to IND-SO-CCA2 secure without chameleon hashing.

Our scheme can be regarded as a variant of the DDH-based construction of Hemenway et al., but free of chameleon hash functions. The scheme has the same efficiency as the original construction: encryption needs $O(n)$ exponentiations, where n is the number of challenge ciphertexts, and a ciphertext consists of 5 elements (4 group elements and one exponent).

2 Notation

Let $\mathcal{H}$ denote a set, $|\mathcal{H}|$ denote the cardinality of the set $\mathcal{H}$, and $h \leftarrow \mathcal{H}$ denote sampling uniformly from the uniform distribution on set $\mathcal{H}$. If $\mathcal{H}$ is a probability distribution, then $h \leftarrow \mathcal{H}$ denotes sampling h according to the distribution. If $A(\cdot)$ is an algorithm, then $a \leftarrow A(\cdot)$ denotes running the algorithm and obtaining a as an output, which is distributed according to the internal randomness of $A(\cdot)$. A function $f(\lambda)$ is *negligible* if for every $c > 0$ there exists an λ_c such that $f(\lambda) < 1/\lambda^c$ for all $\lambda > \lambda_c$.

Let $\mathcal{H}$ be a set of hash functions, mapping $\mathcal{X}$ to $\mathcal{Y}$. Let $k \xleftarrow{\$} \mathbf{Hindex}(1^\lambda)$ denote the index generation algorithm. Each index $k \in \{1, 2, \cdots, |\mathcal{H}|\}$ determines a hash function $H_k \in \mathcal{H}$. Then, $\mathcal{H}$ is collision-resistant if for any polynomial-time adversary $\mathcal{A}$, its advantage $\mathbf{Adv}^{CR}_{\mathcal{H},\mathcal{A}}(\lambda)$, defined as

$$\mathbf{Adv}^{CR}_{\mathcal{H},\mathcal{A}}(\lambda) = \Pr\left[H_k(x_1) = H_k(x_2) : k \xleftarrow{\$} \mathbf{Hindex}(1^\lambda); x_1, x_2 \xleftarrow{\$} \mathcal{A}(H_k)\right],$$

is negligible. $\mathcal{H}$ is *target collision-resistant* (TCR) if for any probabilistic polynomial-time (PPT) adversary $\mathcal{A}$, its advantage $\mathbf{Adv}^{TCR}_{\mathcal{H},\mathcal{A}}(\lambda)$, defined as

$$\mathbf{Adv}^{TCR}_{\mathcal{H},\mathcal{A}}(\lambda) = \Pr\left[H_k(x) = H_k(x'), x \neq x' \mid k \xleftarrow{\$} \mathbf{Hindex}(1^\lambda); x \in \mathcal{X}, x' \xleftarrow{\$} \mathcal{A}(H_k, x)\right],$$

is negligible.

Given a group $\mathbb{G}$ of prime order p, define two distributions $\mathcal{D} = \{(g, g^x, g^y, g^{xy})\}$ and $\mathcal{R} = \{(g, g^x, g^y, g^z)\}$, where $x, y, z \leftarrow \mathbb{Z}_p$. The DDH problem is to distinguish the two distributions. A PPT distinguisher $\mathcal{A}$'s advantage is defined as

$$\mathbf{Adv}^{DDH}_{\mathbb{G},\mathcal{A}}(\lambda) = |\Pr[\mathcal{A}((g, X, Y, Z) \leftarrow \mathcal{D}) = 1] | - \Pr[\mathcal{A}((g, X, Y, Z) \leftarrow \mathcal{R}) = 1] |.$$

The DDH assumption means the advantage of any PPT distinguisher is negligible.

2.1 Chosen-Ciphertext Security (CCA2) in the Selective Opening Setting

Let PKE=(KeyGen, Enc, Dec) be a public key encryption scheme. The key generation algorithm takes as input a security parameter λ, output a public/private key pair, i.e., $(pk, sk) \leftarrow \mathsf{KeyGen}(\lambda)$. The encryption algorithm takes as input a public key pk and a message m and outputs a ciphertext c, i.e., $c \leftarrow \mathsf{Enc}(pk, m)$. The decryption algorithm takes as input a private key sk and a ciphertext c, and outputs a plaintext m or the special symbol $\perp$ meaning that the ciphertext is invalid, i.e., $\{m, \perp\} \leftarrow \mathsf{Dec}(sk, c)$. The above three algorithms are all of polynomial-time. It requires that decryption "undoes" encryption for all $(pk, sk) \leftarrow \mathsf{KeyGen}(\lambda)$ and for $m \in \mathcal{M}$. Here $\mathcal{M}$ denotes the space of plaintexts.

The indistinguishability-based definition of chosen-ciphertext security in the selective opening setting (IND-SO-CCA2) is defined by the following game between an adversary $\mathcal{A}$ and a challenger.

Setup. The challenger obtains a public/private key pair (pk, sk) with $\mathsf{KeyGen}(1^\lambda)$ and gives the public key pk to the adversary.

Decryption Query 1. The adversary $\mathcal{A}$ adaptively issues decryption queries C. The challenger responds with $\mathsf{Dec}(sk, C)$.

Challenge 1. A message sampler $\mathcal{M}$ samples n plaintexts according to a given joint distribution, denoted by $\boldsymbol{m} = (m_1, m_2, \cdots, m_n) \leftarrow \mathcal{M}$. The challenger computes the correponding ciphertexts

$$\begin{aligned}\boldsymbol{C}^* &= (C[1]^*, C[2]^*, \cdots, C[n]^*) \\ &= (\mathsf{Enc}(pk, m_1, r_1), \mathsf{Enc}(pk, m_2, r_2), \cdots, \mathsf{Enc}(pk, m_n, r_n)) .\end{aligned}$$

The challenger sends $\boldsymbol{C}^*$ to the adversary as its challenge vector ciphertext.

Corruption Query. The adversary $\mathcal{A}$ chooses a subset $I \subset \{1, 2, \cdots, n\}$ of size $n/2$, and submits I to the challenger. The challenger then reveals $\{(m_i, r_i)\}_{i \in I}$ to $\mathcal{A}$.

Challenge 2. The challenger picks a random bit $\xi \in \{0, 1\}$. If $\xi = 0$, the challenger sends $\{m_j\}_{j \in \{1,2,\cdots,n\} \setminus I}$ to $\mathcal{A}$. Otherwise, the challenger re-samples $\boldsymbol{m'} = (m'_1, m'_2, \cdots, m'_n) \in \mathcal{M}_{|I, \boldsymbol{m}[I]|}$, i.e., subjected that $m'_j = m_j$ for $j \in I$, and sends $\{m'_j\}_{j \in \{1,2,\cdots,n\} \setminus I}$ to $\mathcal{A}$.

Decryption Query 2. The adversary continues to adaptively issue decryption queries C, as in decryption query phase 1, but with the natural constraint that $C \neq C_i^*$ for $i \in \{1, 2, \cdots, n\}$.

Guess. The adversary $\mathcal{A}$ outputs its guess $\xi' \in \{0, 1\}$ and wins the game if $\xi = \xi'$.

We define $\mathcal{A}$'s advantage in attacking the public key encryption scheme PKE with the security parameter λ as

$$\mathsf{Adv}_{\mathcal{A}}^{\text{IND-SO-CCA2}}(\lambda) = |2\mathsf{Pr}[\xi = \xi'] - 1| = |\Pr[\xi' = 1 | \xi = 1] - \Pr[\xi' = 1 | \xi = 0]|.$$

Definition 1. *We say that a public key encryption scheme PKE is (q, ϵ)-IND-SO-CCA2 secure, if for all PPT algorithms $\mathcal{A}$ with selective opening attack making at most q decryption queries have advantage at most ϵ in winning the above game.*

In [5], the authors proposed the idea of constructing a public-key encryption with IND-SO-CCA2 security. It starts with a tag-based encryption scheme with separable property. Tag-based encryption (TBE) [7,12], denoted by ($\mathsf{TBE.KeyGen}$, $\mathsf{TBE.Enc}$, $\mathsf{TBE.Dec}$), is a public-key encryption where both encryption and decryption take an additional "tag" as input. Separable TBE means a ciphertext consists of three parts $(f_1(pk, m, r), f_2(pk, r), f_3(pk, \theta, r))$, where r is the random coin, θ the tag, and f_1, f_2, f_3 are computed independently of each other. As suggested in [8], a selective-tag weakly secure TBE (i.e., IND-stag-wCCA2) can be transformed into an IND-CCA2 one with help of chameleon hashing. Similarly, the authors in [5] proved that as long as the tag-based encryption schemes in the selective-opening setting satisfies a weaker version of seletive-tag security, i.e. IND-SO-stag-wCCA2, it can also be transform into a full-fledged IND-SO-CCA2 secure one with help of chameleon hashing [5].

A family of chameleon hash functions [21] is a set of randomized collision-resistant (CR) hash functions with an additional property that one can efficiently generate collisions with the help of a trapdoor.

In [5], IND-SO-stag-wCCA2 secure TBE schemes were constructed from all-but-n lossy trapdoor functions. They also showed an IND-SO-stag-wCCA2 scheme constructed directly from the DDH assumption. With help of a chameleon hash function, IND-SO-CCA2 secure PKE can be finally obtained.

In the next section, we will show how to get rid of chameleon hash functions to build a full-fledged IND-SO-CCA2 secure PKE directly from the DDH assumption.

3 Public Key Encryption with IND-SO-CCA2 Security from the DDH Assumption

The PKE consists of three PPT algorithms, (KeyGen, Enc, Dec), as shown below in details.

Key Generation $(pk, sk) \leftarrow \mathbf{KeyGen}(1^\lambda)$**:** On input the security parameter 1^λ, the Key Generation algorithm chooses a group $\mathbb{G}$ of prime order p with two generators g, h. Choose $y \leftarrow \mathbb{Z}_p$ and $a_i, b_i \leftarrow \mathbb{Z}_p$ for $i = 0, 1, \ldots, n$. Let $\boldsymbol{a} = (a_0, a_1, \cdots, a_n)$ and $\boldsymbol{b} = (b_0, b_1, \cdots, b_n)$. Compute $Y = g^y$ and $Y' = h^y$. $A_i = g^{a_i}$, $B_i = g^{b_i}$, $A'_i = h^{a_i}$, and $B'_i = h^{b_i}$ for $i = 0, 1, 2, \ldots, n$. Let $\boldsymbol{A} = (A_0, A_1, \ldots, A_n)$, $\boldsymbol{A'} = (A'_0, A'_1, \ldots, A'_n)$ $\boldsymbol{B} = (B_0, B_1, \ldots, B_n)$ and $\boldsymbol{B'} = (B'_0, B'_1, \ldots, B'_n)$. Choose a collision-resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p$. Set the public key to be $pk = \{\mathbb{G}, g, h, Y, Y', \boldsymbol{A}, \boldsymbol{A'}, \boldsymbol{B}, \boldsymbol{B'}, H\}$ and $sk = \{y, \boldsymbol{a}, \boldsymbol{b}\}$.

Encryption $C \leftarrow \mathbf{Enc}\,(pk, m)$**:** The encryption algorithm encrypts a plaintext m with the public key pk to obtain the corresponding ciphertext C as follows.

1. Choose $r, s \leftarrow \mathbb{Z}_p$ and compute $C_0 = m \cdot Y^r \cdot Y'^s$, $C_1 = g^r \cdot h^s$.
2. Compute $d = H(C_0, C_1)$;
3. Choose $e \leftarrow \mathbb{Z}_p$. Compute $C_2 = \left(A_0^e \prod_{j=1}^n A_j^{d^j}\right)^r \left(A_0'^e \prod_{j=1}^n A_j'^{d^j}\right)^s$ and $C_3 = \left(B_0^e \prod_{j=1}^n B_j^{d^j}\right)^r \left(B_0'^e \prod_{j=1}^n B_j'^{d^j}\right)^s$;
4. Set $C = (C_0, C_1, C_2, C_3, e)$.

Decryption $m \leftarrow \mathbf{Dec}\,(sk, C)$**:** The decryption algorithm decrypts a ciphertext $C = (C_0, C_1, C_2, C_3, e)$ with the secret key $sk = \{y, \boldsymbol{a}, \boldsymbol{b}\}$ to obtain the corresponding plaintext m as follows.

1. The consistency of the ciphertext is checked by

$$C_2 = C_1^{a_0 e + \sum_{j=1}^n a_j d^j}, \quad C_3 = C_1^{b_0 e + \sum_{j=1}^n b_j d^j}, \tag{1}$$

   where $d = H(C_0, C_1)$. If Eq.(1) does not hold, output $\perp$, and go to the next step otherwise.
2. Compute $m = C_0 / C_1^y$.

Each ciphertext C consists of five elements. The first four elements are all from $\mathbb{G}$ and the last one is from $\mathbb{Z}_p$. The original IND-SO-stag-wCCA2 scheme in [5] consists of 4 elements of $\mathbb{G}$ in a ciphertext. When it is changed into an IND-SO-CCA2 one, the ciphertext is added with one more element, which is one of the two inputs of the chameleon hash function. Therefore, the size of ciphertexts in our scheme is comparable to that in [5]. Efficiency of encryption and decryption is also comparable to the scheme in [5], but our scheme does not need chameleon hash functions at all.

The difference between our scheme and the IND-SO-stag-wCCA2 scheme in [5] is the choice of the tag. The tag in the IND-SO-stag-wCCA2 scheme in [5] is just one element, and that is why only weak selective-tag security is achieved. The tag in our scheme consists of two parts, one is determined by the ciphertext and the other is randomly chosen. The free choice of the second part of tag makes possible that the challenge ciphertext distributed properly when the DDH problem is embedded into the PKE, while the ciphertexts queried by the adversary are still able to be correctly decrypted. Similar proof technique based on flexible tags was also implemented in [22] to achieve IND-CCA2 security.

Theorem 1. *The above public key encryption scheme is* (q, ϵ)*-IND-SO-CCA2 secure, where* q *is the number of decryption queries and*

$$\epsilon \leq \mathbf{Adv}^{\mathbf{TCR}}_{\mathcal{H},\mathcal{A}'}(\lambda) + \mathbf{Adv}^{\mathbf{DDH}}_{\mathbb{G},\mathcal{A}''}(\lambda) + \mathbf{q}/\mathbf{2}^{\boldsymbol{\lambda-1}}.$$

Proof. To prove the theorem, all we have to do is to prove the following holds

$$\mathbf{Adv}^{\text{IND-SO-CCA2}}_{\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathbf{TCR}}_{\mathcal{H},\mathcal{A}'}(\lambda) + \mathbf{Adv}^{\mathbf{DDH}}_{\mathbb{G},\mathcal{A}''}(\lambda) + \mathbf{q}/\mathbf{2}^{\boldsymbol{\lambda-1}}.$$

We proceed with a series of games played between a simulator $\mathcal{D}$ and an adversary $\mathcal{A}$, and show that Game i and Game $i+1$ are indistinguishable except with negligible probability, $i = 0, 1, 2, 3, 4, 5, 6$. We define S_i as the event that the adversary $\mathcal{A}$ output 1.

Game 0: The simulator generates the public/secret key with $(pk, sk) \leftarrow \mathsf{KeyGen}(1^\lambda)$, where $pk = \{\mathbb{G}, g, h, Y, Y', \boldsymbol{A}, \boldsymbol{A'}, \boldsymbol{B}, \boldsymbol{B'}, H\}$ and $sk = \{y, \boldsymbol{a}, \boldsymbol{b}\}$. The simulator sends pk to the adversary. For each decryption query $C = (C_0, C_1, C_2, C_3, e)$ made by $\mathcal{A}$, the simulator returns $m \leftarrow \mathsf{Dec}(sk, C)$ using the secret key $sk = \{y, \boldsymbol{a}, \boldsymbol{b}\}$, where $\boldsymbol{a} = (a_0, a_1, \cdots, a_n)$ and $\boldsymbol{b} = (b_0, b_1, \cdots, b_n)$. After decryption queries of polynomially times, the simulator chooses a vector of n plaintexts $\boldsymbol{m^*} = (m_1^*, m_2^*, \cdots, m_n^*)$ from $\mathcal{M}$, and computes the corresponding ciphertext vector $\boldsymbol{C^*} = (C[1]^*, C[2]^*, \cdots, C[n]^*)$ with $C[i]^* = (C^*_{i,0}, C^*_{i,1}, C^*_{i,2}, C^*_{i,3}, C^*_{i,4})$ in the following way. Choose $r_i^*, s_i^*, e_i^* \in \mathbb{Z}_p$ and compute

$C^*_{i,0} = m_i^* \cdot Y^{r_i^*} \cdot Y'^{s_i^*}$,
$C^*_{i,1} = g^{r_i^*} \cdot h^{s_i^*}$,
$d_i^* = H(C^*_{i_0}, C^*_{i,1})$,
$C^*_{i,2} = \left(A_0^{e_i^*} \prod_{j=1}^n A_j^{d_i^{*j}}\right)^{r_i^*} \cdot \left(A_0'^{e_i^*} \prod_{j=1}^n A_j'^{d_i^{*j}}\right)^{s_i^*}$,

$C^*_{i,3} = \left(B_0^{e_i^*} \cdot \prod_{j=1}^n B_j^{d_i^{*j}}\right)^{r_i^*} \cdot \left(B_0'^{e_i^*} \prod_{j=1}^n B_j'^{d_i^{*j}}\right)^{s_i^*}$,
$C^*_{i,4} = e_i^*$.

The simulator $\mathcal{D}$ gives the ciphertext vector $\boldsymbol{C^*} = (C[1]^*, C[2]^*, \cdots, C[n]^*)$ to the adversary $\mathcal{A}$. $\mathcal{A}$ continues to ask decryption queries. When $\mathcal{A}$ makes a corruption query, $\mathcal{A}$ chooses a subset $I \subseteq \{1, 2, \cdots, n\}$ such that $|I| = n/2$ and submits I to $\mathcal{D}$. $\mathcal{D}$ returns $\{m_i^*, r_i^*, s_i^*\}$ for all $i \in I$. The challenger also sends $\{m_j^*\}_{j \in \{1,2,\cdots,n\} \setminus I}$ to $\mathcal{A}$.

Let S_0 be the event that $\mathcal{A}$ outputs 1 at the end of the game. Then $\Pr[S_0] = \Pr[\xi' = 1 | \xi = 0]$.

Game 1: is the same as Game 0 except for the public/secret key generation. In this game, the simulator $\mathcal{D}$ generate the public key

$$pk = \{G, g, h, Y, Y', \boldsymbol{A}, \boldsymbol{A'}, \boldsymbol{B}, \boldsymbol{B'}, H\}$$

as follows.

- Choose $x \leftarrow \mathbb{Z}_p$, and compute $X = g^x, X' = h^x$.
- Choose $w_1, w_2 \leftarrow \mathbb{Z}_p$ and compute $Y = g^{w_1} X^{w_2}, Y' = h^{w_1} X'^{w_2}$.
- Choose $\alpha_i, \beta_i, \gamma_i \leftarrow \mathbb{Z}_p$ for $i = 0, 1, \cdots, n$.
  Compute $A_i = X^{\alpha_i} g^{\beta_i}, A'_i = X'^{\alpha_i} h^{\beta_i}$, $B_i = Y^{\alpha_i} g^{\gamma_i}, B'_i = Y'^{\alpha_i} h^{\gamma_i}$.
  Let $\boldsymbol{A} = (A_0, A_1, \ldots, A_n)$, $\boldsymbol{A'} = (A'_0, A'_1, \ldots, A'_n)$, $\boldsymbol{B} = (B_0, B_1, \ldots, B_n)$ and $\boldsymbol{B'} = (B'_0, B'_1, \ldots, B'_n)$.

Note that the corresponding secret key $sk = (y, \boldsymbol{a}, \boldsymbol{b})$ is given by $y = w_1 + w_2 x, a_i = \alpha_i x + \beta_i, b_i = \alpha_i y + \gamma_i$ for $i = 0, 1, \cdots, n$.

To decrypt a ciphertext $C = (C_0, C_1, C_2, C_3, e)$, the simulator $\mathcal{D}$ checks that

$$C_2 = C_1^{(\alpha_0 x + \beta_0)e + \sum_{j=1}^n (\alpha_j x + \beta_j) d^j}, \quad C_3 = C_1^{(\alpha_0 y + \gamma_0)e + \sum_{j=1}^n (\alpha_j y + \gamma_j) d^j} \tag{2}$$

hold, then responses with $m = C_0 / C_1^{w_1 + w_2 x}$.

The distribution of the public key pk is exactly the same as that in Game 0. Hence $\Pr[S_1] = \Pr[S_0]$.

Game 2: is the same as Game 1 except for the generation of the challenge vector ciphertext. The simulator $\mathcal{D}$ chooses r_i^*, s_i^* for $i = 1, 2, \ldots, n$ and computes

$C^*_{i,0} = m_i^* \cdot Y_1^{r_i^*} \cdot Y_2^{s_i^*}$,
$C^*_{i,1} = g^{r_i^*} \cdot h^{s_i^*}$,
$d_i^* = H(C^*_{i0}, C^*_{i,1})$,
computes $e_i^* = C^*_{i,4} = -\alpha_0^{-1} \cdot \left(\alpha_1 d_i^* + \alpha_1 d_i^{*2} + \cdots + \alpha_n d_i^{*n}\right)$.
$C^*_{i,2} = \left(A_0^{e_i^*} \prod_{j=1}^n A_j^{d_i^{*j}}\right)^{r_i^*} \cdot \left(A_0'^{e_i^*} \prod_{j=1}^n A_j'^{d_i^{*j}}\right)^{s_i^*}$,
$C^*_{i,3} = \left(B_0^{e_i^*} \cdot \prod_{j=1}^n B_j^{d_i^{*j}}\right)^{r_i^*} \cdot \left(B_0'^{e_i^*} \prod_{j=1}^n B_j'^{d_i^{*j}}\right)^{s_i^*}$.

Let $Q(T) = \alpha_0 e_i^* + \alpha_1 T + \alpha_1 T^2 + \cdots + \alpha_n T^n$, $Q_2(T) = \beta_0 e_i^* + \beta_1 T + \beta_1 T^2 + \cdots + \beta_n T^n$ and $Q_3(T) = \gamma_0 e_i^* + \gamma_1 T + \gamma_1 T^2 + \cdots + \gamma_n T^n$. Then

$$C^*_{i,2} = \left(g^{Q_2(d_i^*)} X^{Q(d_i^*)}\right)^{r_i^*} \left(h^{Q_2(d_i^*)} X'^{Q(d_i^*)}\right)^{s_i^*} = \left(g^{Q_2(d_i^*)}\right)^{r_i^*} \left(h^{Q_2(d_i^*)}\right)^{s_i^*}$$

$$C^*_{i,3} = \left(g^{Q_3(d_i^*)} Y^{Q(d_i^*)}\right)^{r_i^*} \left(h^{Q_3(d_i^*)} Y'^{Q(d_i^*)}\right)^{s_i^*} = \left(g^{Q_3(d_i^*)}\right)^{r_i^*} \left(h^{Q_3(d_i^*)}\right)^{s_i^*},$$

since $Q(d_i^*) = \alpha_0 e_i^* + \alpha_1 d_i^* + \alpha_1 d_i^{*2} + \cdots + \alpha_n d_i^{*n} = 0$.
It does not change the distribution of the challenge vector ciphertext, hence $\Pr[S_2] = \Pr[S_1]$.

Game 3: is the same as Game 2 except that the simulator $\mathcal{D}$ applies a special rejection rule. If $\mathcal{A}$ asks for a decryption of $C = (C_0, C_1, C_2, C_3, e)$ such that $(C_0, C_1) \neq (C^*_{i,0}, C^*_{i,1})$ but $H(C_0, C_1) = d_i^* = H(C^*_{i,0}, C^*_{i,1})$ for some $i \in \{1, 2, \cdots, n\}$, the simulator $\mathcal{D}$ rejects with $\perp$ and the game aborts. Let F be the event that $\mathcal{D}$ outputs $\perp$, then $\Pr[S_2|\neg F] = \Pr[S_1|\neg F]$ and $|\Pr[S_2] - \Pr[S_1]| \leq \Pr[F]$. The event F occurs with non-negligible probability, otherwise we can construct an algorithm $\mathcal{A}'$ to contradict the collision resistance property of H. More precisely,

$$|\Pr[S_3] - \Pr[S_2]| \leq \Pr[F] \leq \mathbf{Adv}^{\mathrm{TCR}}_{\mathcal{H},\mathcal{A}'}(\lambda).$$

Game 4: is the same as Game 3 except for the behavior of decryption oracle supplied by $\mathcal{D}$. For a decryption of $C = (C_0, C_1, C_2, C_3, e)$ asked by $\mathcal{A}$, $\mathcal{D}$ computes $d = H(C_0, C_1)$. If $Q(d) = \alpha_0 e + \alpha_1 d + \alpha_1 d^2 + \cdots + \alpha_n d^n = 0$, the simulator $\mathcal{D}$ outputs $\perp$ and the game aborts. Let F' be the event that $\mathcal{D}$ outputs $\perp$ for $Q(d) = 0$. Then $\Pr[S_4|\neg F'] = \Pr[S_3|\neg F']$ and

$$|\Pr[S_4] - \Pr[S_3]| \leq \Pr[F'].$$

Now we evaluate the upper bound of $\Pr[F']$. To the adversary, α_i, $i = 0, 1, \cdots, n$, is uniformly distributed in $\mathbb{Z}_p$ given the public key. The challenge vector ciphertext leaks information about α_i by the following equations

$$\begin{pmatrix} e_1^* & d_1^* & \cdots & d_1^{*n} \\ e_2^* & d_2^* & \cdots & d_2^{*n} \\ \vdots & \vdots & \cdots & \vdots \\ e_n^* & d_n^* & \cdots & d_n^{*n} \end{pmatrix} \cdot \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

The size of solution space for $(\alpha_0, \alpha_1, \cdots, \alpha_n)$ is p. The additional equation $Q(d) = 0$ will fix a unique solution among the p choices, which implies $\Pr[Q(d) = 0] = 1/p$.
Therefore

$$|\Pr[S_4] - \Pr[S_3]| \leq \Pr[F'] \leq q/p \leq q/2^{\lambda},$$

with q the number of queries.

Game 5: is the same as Game 4 except for the behavior of decryption oracle supplied by $\mathcal{D}$. For a decryption of $C = (C_0, C_1, C_2, C_3, e)$ asked by $\mathcal{A}$, $\mathcal{D}$ computes $d = H(C_0, C_1)$, then evaluates three polynomials $Q(T) = \alpha_0 e + \alpha_1 T + \alpha_1 T^2 + \cdots + \alpha_n T^n$, $Q_2(T) = \beta_0 + \beta_1 T + \beta_1 T^2 + \cdots + \beta_n T^n$ and $Q_3(T) = \gamma_0 + \gamma_1 T + \gamma_1 T^2 + \cdots + \gamma_n T^n$ at d. $\mathcal{D}$ computes

$$Z_i = \left(C_i / C_1^{Q_i(d)}\right)^{1/Q(d)}$$

for $i = 2, 3$. Check whether

$$C_1^{w_1} Z_2^{w_2} = Z_3 \tag{3}$$

or not. If not, the simulator $\mathcal{D}$ regards C as an invalid ciphertext and rejects with $\perp$. Otherwise, $\mathcal{D}$ responds with $m = C_0/Z_3$ to $\mathcal{A}$. Note that now $\mathcal{D}$ does not need x for decryption any more. This consistency check uses the idea of "Twin Diffie-Hellman trapdoor test" [13]. Just like [5], we show that the probability that an invalid ciphertext passes the trapdoor test of Eq. (3) with negligible probability.
Define polynomials

$$A(T) = (\alpha_0 ex + \beta_0) + (\alpha_1 x + \beta_1)T + \cdots + (\alpha_n x + \beta_n)T^n$$

and

$$B(T) = (\alpha_0 ey + \gamma_0) + (\alpha_1 y + \gamma_1)T + \cdots + (\alpha_n y + \gamma_n)T^n.$$

Then $A(T) = Q(T)x + Q_2(T)$ and $B(T) = Q(T)y + Q_3(T)$. A ciphertext $C = (C_0, C_1, C_2, C_3, e)$ is consistent if

$$C_1 = g^r h^s, \;\; C_2 = C_1^{A(d)} = C_1^{Q_2(d)+xQ(d)}, \;\; C_3 = C_1^{B(d)} = C_1^{Q_3(d)+yQ(d)}. \tag{4}$$

Define $\tau = r + s \log_g h$, then a consistent ciphertext satisfies

$$C_1 = g^\tau, \quad C_2 = g^{(Q_2(d)+xQ(d))\tau}, \quad C_3 = g^{(Q_3(d)+yQ(d))\tau}. \tag{5}$$

An inconsistent ciphertext $(\widetilde{C_0}, \widetilde{C_1}, \widetilde{C_2}, \widetilde{C_3})$ can similarly expressed as

$$\widetilde{C_1} = g^\tau, \quad \widetilde{C_2} = g^{t_2}, \quad \widetilde{C_3} = g^{t_3}, \tag{6}$$

where either $t_2 \neq (Q_2(d) + xQ(d))\tau$ or $t_3 \neq (Q_3(d) + yQ(d))\tau$.
The trapdoor test computes

$$\widetilde{Z_i} = \left(\widetilde{C_i}/\widetilde{C_1}^{Q_i(d)}\right)^{1/Q(d)}$$

for $i = 2, 3$. Let $\widetilde{Z_2} = g^{x\tau_2}, \widetilde{Z_3} = g^{y\tau_3}$. Then either $\tau_2 \neq \tau$ or $\tau_3 \neq \tau$.
If $\tau_2 = \tau$ but $\tau_3 \neq \tau$, then this ciphertext will never pass the trapdoor test of Eq. (3). Therefore we assume $\tau_2 \neq \tau$. If this inconsistent ciphertext passes the trapdoor test of Eq(3), we have

$$\begin{pmatrix} 1 & x \\ \tau & \tau_2 x \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} y \\ \tau_3 y \end{pmatrix}. \tag{7}$$

The matrix $\begin{pmatrix} 1 & x \\ \tau & \tau_2 x \end{pmatrix}$ is of full rank, due to $\tau \neq \tau_2$. Fixed x, y, τ, τ_2, each τ_3 determines a unique solution (w_1, w_2). As to the adversary $\mathcal{A}$, the random variable (w_1, w_2) are uniformly distributed, that means τ_3, hence $\tau_3 y$, is also

uniformly distributed. Therefore, the trapdoor test $\widetilde{C_1}^{w_1}\widetilde{Z_2}^{w_2} = \widetilde{Z_3}$ holds with probability $1/p$.

Let F'' be the event that there exists an inconsistent ciphertext satisfies the trapdoor test among the adversary's q queries. i.e., $\mathcal{D}$ accepts the ciphertext even if it is an inconsistent one. Then $\Pr[S_5|\neg F''] = \Pr[S_4|\neg F'']$ and

$$|\Pr[S_5] - \Pr[S_4]| \leq \Pr[F''] \leq q/p \leq q/2^{\lambda},$$

where q is the number of queries.

Game 6: is the same as Game 5 except for the generation of public key pk. The only difference is that the simulator $\mathcal{D}$ chooses $X = g^x$ but $X' = h^{x'}$ with $x' \leftarrow \mathbb{Z}_p \setminus \{x\}$ in the generation of pk. $\mathcal{D}$ is still able to answer corruption queries with $\{m_i^*, r_i^*, s_i^*\}$ and answer decryption queries exactly like that in Game 5.

Note that (g, h, X, X') is independent of the rest of the public key. The tuple $(g, h, X = g^x, X' = h^x)$ in Game 5 is DDH tuple, while $(g, h, X = g^x, X' = h^{x'})$ is a random tuple in $\mathbb{G}$. Any non-negligible difference between Game 6 and 5 results in a DDH-adversary $\mathcal{A}''$ with advantage $\mathbf{Adv}^{\mathrm{DDH}}_{\mathbb{G},\mathcal{A}''}(\lambda) = |\Pr[S_6] - \Pr[S_5]|$.

Now we show that $\Pr[S_6] = \Pr[\xi' = 1|\xi = 1]$.

The challenge vector ciphertext is $\left(C_{i,0}^*, C_{i,1}^*, C_{i,2}^*, C_{i,3}^*, e_i^*\right)$ with $i = 1, 2, \cdots, n$, where

- $C_{i,0}^* = m_i^* \cdot Y^{r_i^*} \cdot Y'^{s_i^*}$,
- $C_{i,1}^* = g^{r_i^*} \cdot h^{s_i^*}$,
- $C_{i,2}^* = \left(g^{Q_2(d_i^*)}\right)^{r_i^*} \left(h^{Q_2(d_i^*)}\right)^{s_i^*} = \left(g^{r_i^*} h^{s_i^*}\right)^{Q_2(d_i^*)}$,
- $C_{i,3}^* = \left(g^{Q_3(d_i^*)}\right)^{r_i^*} \left(h^{Q_3(d_i^*)}\right)^{s_i^*} = \left(g^{r_i^*} h^{s_i^*}\right)^{Q_3(d_i^*)}$.

Let $\delta = \log_g(C_{i,0}^*) - \log_g m_i^*$, $y = \log_g Y$ and $y' = \log_g Y'$. According to the information the adversary $\mathcal{A}$ obtained from the challenger ciphertext C_i^*, let us see what $\mathcal{A}$ knows about δ. This equation

$$\begin{pmatrix} 1 & \log_g h \\ Q_2(d_i^*) & Q_2(d_i^*)\log_g h \\ Q_3(d_i^*) & Q_3(d_i^*)\log_g h \\ y & y' \end{pmatrix} \cdot \begin{pmatrix} r_i^* \\ s_i^* \end{pmatrix} = \begin{pmatrix} \log_g C_{i,1}^* \\ \log_g C_{i,2}^* \\ \log_g C_{i,3}^* \\ \delta \end{pmatrix} \tag{8}$$

can be reduced to

$$\begin{pmatrix} 1 & \log_g h \\ y & y' \end{pmatrix} \cdot \begin{pmatrix} r_i^* \\ s_i^* \end{pmatrix} = \begin{pmatrix} \log_g C_{i,1}^* \\ \delta \end{pmatrix}, \tag{9}$$

since the second and third row of the 4 by 2 matrix is linearly dependent to the first one.

We know that $y = w_1 + xw_2$ and $y' = w_1 + x'w_2$, hence the matrix $\begin{pmatrix} 1 & \log_g h \\ y & y' \end{pmatrix}$ is of full rank. The random variables (r_i^*, s_i^*) are uniformly distributed as to $\mathcal{A}$. Then each specific value of δ determine a unique solution (r_i^*, s_i^*).

Therefore, δ is uniformly distributed in $\mathbb{Z}_p$. That means $m_i^* = C_{i,0}^*/g^\delta$ is also uniformly distributed over $\mathbb{G}$ for $i \in \{0, 1, \cdots, n-1\}$. That is exactly the case that the challenger choose $\xi = 1$. That explains why $\Pr[S_6] = \Pr[\xi' = 1 | \xi = 1]$.

4 Conclusion

In this paper, we showed how to build a public-key encryption scheme with IND-SO-CCA2 security directly from the DDH Assumption. Our work is based on the IND-stag-SO-wCCA2 secure scheme in [5]. In [5], Hemenway, Libert, Ostrovsky and Vergnaud provided definitions of adaptive chosen-ciphertext security in selective opening setting, and proposed a paradigm to construct encryption schemes achieving the defined security. They start with a weak selective-tag secure tag-based encryption and apply a chameleon hash to change it to a scheme with full-fledged security. We proposed to use two elements to make up a tag. The choice of tags is much flexible, and this serves proof of IND-SO-CCA2 security very well. Similar techniques can also be applied to Paillier-based scheme in [5]. Up to now Hemenway et.al.'s method of constructing IND-SO-CCA2 PKE from an IND-stag-SO-wCCA2 one is the only general construction. The open question is whether we can find a new paradigm free of chameleon hashing.

References

1. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively Secure Multi-Party Computation. In: STOC 1996, pp. 639–648. ACM, New York (1996)
2. Bellare, M., Yilek, S.: Encryption Schemes Secure under Selective Opening Attack. Cryptology ePrint Archive: Report 2009/101 (2009)
3. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and Impossibility Results for Encryption and Commitment Secure under Selective Opening. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 1–35. Springer, Heidelberg (2009)
4. Hofheinz, D.: Possibility and Impossibility Results for Selective Decommitments. Cryptology ePrint Archive, Report 2008/168, `http://eprint.iacr.org/2008/168`
5. Hemenway, B., Libert, B., Ostrovsky, R., Vergnaud, D.: Lossy Encryption: Constructions from General Assumptions and Efficient Selective Opening Chosen-Ciphertext Security. In: Lee, D.H. (ed.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 70–88. Springer, Heidelberg (2011)
6. Boldyreva, A., Fehr, S., O'Neill, A.: On Notions of Security for Deterministic Encryption, and Efficient Constructions without Random Oracles. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 335–359. Springer, Heidelberg (2008)
7. MacKenzie, P.D., Reiter, M.K., Yang, K.: Alternatives to Non-malleability: Definitions, Constructions, and Applications. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 171–190. Springer, Heidelberg (2004)
8. Zhang, R.: Tweaking TBE/IBE to PKE Transforms with Chameleon Hash Functions. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 323–339. Springer, Heidelberg (2007)
9. Boyen, X., Waters, B.: Shrinking the Keys of Discrete-Log-Type Lossy Trapdoor Functions. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 35–52. Springer, Heidelberg (2010)

10. Damgård, I., Jurik, M.: A Generalisation, A Simplification and Some Applications of Paillier's Probabilistic Public-Key System. In: Kim, K.-C. (ed.) PKC 2001. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
11. Freeman, D.M., Goldreich, O., Kiltz, E., Rosen, A., Segev, G.: More Constructions of Lossy and Correlation-Secure Trapdoor Functions. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 279–295. Springer, Heidelberg (2010)
12. Kiltz, E.: Chosen-Ciphertext Security from Tag-Based Encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006)
13. Cash, D., Kiltz, E., Shoup, V.: The Twin Diffie-Hellman Problem and Applications. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 127–145. Springer, Heidelberg (2008)
14. Fehr, S., Hofheinz, D., Kiltz, E., Wee, H.: Encryption Schemes Secure against Chosen-Ciphertext Selective Opening Attacks. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 381–402. Springer, Heidelberg (2010)
15. Bellare, M., Dowsley, R., Waters, B., Yilek, S.: Standard Security Does Not Imply Security Against Selective-Opening. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 645–662. Springer, Heidelberg (2012)
16. Böhl, F., Hofheinz, D., Kraschewski, D.: On Definitions of Selective Opening Security. In: Fischlin, M., Buchmann, J., Manulis, M. (eds.) PKC 2012. LNCS, vol. 7293, pp. 522–539. Springer, Heidelberg (2012)
17. Hemenway, B., Ostrovsky, R.: Lossy Trapdoor Functions from Smooth Homomorphic Hash Proof Systems. In: Electronic Colloquium on Computational Complexity (ECCC 2009), p. 127 (2009)
18. Hemenway, B., Ostrovsky, R.: Homomorphic Encryption Over Cyclic Groups Implies Chosen-Ciphertext Security. Cryptology ePrint Archive, Report 2010/099 (2010)
19. Hofheinz, D.: All-But-Many Lossy Trapdoor Functions. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 209–227. Springer, Heidelberg (2012)
20. Kiltz, E., Mohassel, P., O'Neill, A.: Adaptive Trapdoor Functions and Chosen-Ciphertext Security. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 673–692. Springer, Heidelberg (2010)
21. Krawczyk, H., Rabin, T.: Chameleon Signatures. In: NDSS 2000, pp.143-154. The Internet Society (2000)
22. Lai, J., Deng, R.H., Liu, S.: Chameleon All-But-One TDFs and Their Application to Chosen-Ciphertext Security. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 228–245. Springer, Heidelberg (2011)
23. Mohassel, P.: One-time Signatures and Chameleon Hash Functions. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 302–319. Springer, Heidelberg (2011)
24. Peikert, C., Waters, B.: Lossy Trapdoor Functions and Their Applications. SIAM J. Comput. 40(6), 1803–1844 (2011)
25. Rosen, A., Segev, G.: Chosen-Ciphertext Security via Correlated Products. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 419–436. Springer, Heidelberg (2009)
26. Shamir, A., Tauman, Y.: Improved Online/Offline Signature Schemes. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 355–367. Springer, Heidelberg (2001)

Proxy Signature Scheme Based on Isomorphisms of Polynomials

Shaohua Tang and Lingling Xu

School of Computer Science & Engineering,
South China University of Technology, Guangzhou, China
csshtang@scut.edu.cn, shtang@IEEE.org

Abstract. The proxy signatures are important cryptosystems that are widely adopted in different applications. Most of the proxy signature schemes so far are based on the hardness of integer factoring, discrete logarithm, and/or elliptic curve. However, Shor proved that the emerging quantum computers can solve the problem of prime factorization and discrete logarithm in polynomial-time, which threatens the security of current RSA, ElGamal, ECC, and the proxy signature schemes based on these problems. We propose a novel proxy signature scheme based on the problem of Isomorphism of Polynomials (IP) which belongs to a major category of Multivariate Public Key Cryptography (MPKC). Through security discussion, our scheme can reach the same security level as the signature scheme based on IP problem. The most attractive advantage of our scheme should be its feature to potentially resist the future quantum computing attacks. Our scheme also owns some important properties of proxy signature schemes, such as strong unforgeability, strong identifiability, strong undeniability, secret-key's dependence, distinguishability, etc. The scheme is implemented in C/C++ programming language, and the performance shows that the scheme is efficient. The parameters we choose can let security level of the scheme up to $2^{86.59}$.

Keywords: Post-Quantum Cryptography, Multivariate Public Key Cryptography, Isomorphism of Polynomials, Proxy Signature, Digital Signature.

1 Introduction

A proxy signature protocol allows an entity, called original signer, to delegate another entity, called a proxy signer, to sign messages on behalf of the original signer. The first efficient proxy signature was introduced by Mambo, Usuda and Okamoto [12, 13]. The types of delegation can be classified into full delegation, partial delegation, delegation by warrant, and partial delegation with warrant [11]. A considerable number of proxy signature schemes have been constructed for each of these delegation types, as shown in [3]. Proxy signatures have found numerous practical applications, particularly in distributed computing where delegation of rights is quite common, for example, grid computing, mobile agent applications, and mobile communications. The basic proxy signature has been

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 113–125, 2012.

extended to own various features, for example, threshold proxy signatures [20], blind proxy signatures [1], anonymous proxy signatures [8], etc.

Almost all the proxy signature schemes so far are based on the difficulty problem of integer factoring, discrete logarithm, and/or elliptic curve. However, Shor [18] proved that the emerging quantum computers can solve the problem of prime factorization and discrete logarithm in polynomial-time, which threatens the security of current RSA, ElGamal, DSA, ECC, and the proxy signature schemes based on these problems.

In order to resist the attacks of quantum computing, the Post-Quantum Cryptography has attracted cryptographers' intensive attentions. Some cryptosystems, such as hash-based cryptography, coding-based cryptography, lattice-based cryptography, and Multivariate Public Key Cryptography (MPKC), belong to the area of Post-Quantum Cryptography [2].

The security of MPKC is based on the hardness of solving a set of multivariate polynomial equations over a finite field, which is proven to be an NP-hard problem [9], and the quantum computers do not appear to have any advantages when dealing with this NP-hard problems. Cryptosystem based on the problem of Isomorphism of Polynomials (IP) is a major category of MPKC. Therefore, our proposed proxy signature scheme based on IP problem has the potential advantage of resisting the attacks of future quantum computers.

Our Contribution. After the introduction of the problem of Isomorphism of Polynomials, we simplify Patarin's signature algorithm [15] based on IP problem to become a compact and workable digital signature scheme that we call IP signature.

Then we propose a proxy signature scheme based on IP signature, which includes the stages of initialization, delegation and proxy key generation, generation of proxy signature, and the verification of proxy signature.

After that, security analysis shows that our proxy signature scheme shares the same security level with the underlying IP signature scheme. Some important properties of proxy signature schemes, such as strong unforgeability, strong identifiability, strong undeniability, secret-key's dependence, distinguishability, etc., are also owned by our scheme.

Finally, we implement the scheme in C/C++ programming language. The performance shows that the scheme can run efficiently, and the chosen parameters can let the security level of the scheme up to $2^{86.59}$.

Organization. The rest of the paper is organized as follows. In Section 2, we introduce the problem of Isomorphisms of Polynomials and IP signature scheme. Then our proposed proxy signature scheme based on IP signature is described in Section 3. We discuss the security of our scheme in Section 4. The implementation and performance of our scheme are described in Section 5. Finally, the conclusion is summarized in Section 6.

2 Preliminaries

Some basic building blocks adopted by our proposed scheme are introduced in this section, which includes the problem of Isomorphism of Polynomials (IP), the signature algorithm based on IP, and the procedure to verify the IP signature.

2.1 Problem of Isomorphism of Polynomials

The problem of Isomorphism of Polynomials was introduced by Patarin [15]. It is a fundamental problem of multivariate cryptography, since it is related to the hardness of the key recovery of such cryptosystems. The concept of IP is briefly described as follows. For more details, we refer the reader to [15].

Let K be a finite field, and all the arithmetic operations are over this field. Let n and u be positive integers. Let A be a set of u quadratic equations with n variables $x_1, \cdots, x_n$ that give the y values from the x values:

$$y_k = \sum_i \sum_j \gamma_{ijk} x_i x_j + \sum_i \mu_{ik} x_i + \delta_k, \quad \text{for } k = 1, \cdots, u. \tag{1}$$

Let B be a set of u quadratic equations with n variables $x'_1, \cdots, x'_n$ that give the y' values from the x' values:

$$y'_k = \sum_i \sum_j \gamma'_{ijk} x'_i x'_j + \sum_i \mu'_{ik} x'_i + \delta'_k, \quad \text{for } k = 1, \cdots, u. \tag{2}$$

Let S be a bijective affine transformation of the variables $y_k, \ldots, y_u$, which is defined by

$$S(y_1, \cdots, y_u) = (y'_1, \cdots, y'_u); \tag{3}$$

and T be a bijective affine transformation of the variables $x'_i, \ldots, x'_n$, which is defined by

$$T(x'_1, \cdots, x'_n) = (x_1, \cdots, x_n). \tag{4}$$

If there exists such transformation pair (S, T) which satisfies $B = S \circ A \circ T$, then we call A and B are *"isomorphic"*, and the bijective affine transformation pair (S, T) is an *"isomorphism"* from A to B.

Definition 1. (IP Problem) *The* Problem of Isomorphism of Polynomials *(abbreviated* IP Problem*) is the problem to find an isomorphism* (S, T) *from* A *to* B*, where* A *and* B *are two public sets of* u *quadratic equations, and* A *and* B *are isomorphic.*

2.2 IP Signature

Zero-knowledge Proofs of Knowledge allow a prover to demonstrate the knowledge of a secret without leaking any useful information about the secret. Patarin [15] presented a Non-interactive Zero-knowledge Proofs of knowledge scheme based on the Problem of Isomorphism of Polynomials. We simplify Patarin's algorithm to become a compact and workable digital signature scheme that we call *IP signature*, which is described as follows.

Generation of IP Signature. Let K be a finite field, and n, u and q be positive integers. Let $\mathcal{H}$ be a collision-resistant hash function

$$\mathcal{H} : \{0,1\}^* \to \{0,1\}^q,$$

which maps a binary string of arbitrary length to a q-bit hash value.

Let $H = \mathcal{H}(\cdot)$, and we denote the i-th bit of H value by $H[i]$, where $H[i] \in \{0,1\}$, and $i = 1, \ldots, q$; then the binary format of value H can be represented as $H[q] \ldots H[2]H[1] \in \{0,1\}^q$. Let A and B be two sets of u quadratic equations with n variables, which are defined in (1) and (2), respectively; and S and T be two bijective affine transformations defined in (3) and (4) respectively, and (S,T) be an isomorphism from A to B.

Suppose that Alice is a signer whose private key is (S,T), and its corresponding public key is (A,B). Let m be the message to sign. Alice can sign a message based on the IP problem by invoking the algorithm **IPSign**, which is described as follows.

Algorithm 1. **IPSign(** $m, (S,T), (A,B)$ **)**
// to generate a digital signature on message based on IP problem;

Input

m: the message to sign;
(S,T): the private key to sign the message;
(A,B): the public key corresponding to (S,T);

Output

V: the signature on message m;

Begin

Step 1. The signer selects q random bijective affine transformation pairs $(S'_1, T'_1), (S'_2, T'_2), \cdots, (S'_q, T'_q)$, which are in the following form:

$$\begin{aligned}
&S'_1(y_1, \cdots, y_u) = (y_1^{(1)}, \cdots, y_u^{(1)}),\\
&\cdots\cdots\\
&S'_q(y_1, \cdots, y_u) = (y_1^{(q)}, \cdots, y_u^{(q)}),\\
&T'_1(x_1^{(1)}, \cdots, x_n^{(1)}) = (x_1, \cdots, x_n),\\
&\cdots\cdots\\
&T'_q(x_1^{(q)}, \cdots, x_n^{(q)}) = (x_1, \cdots, x_n);
\end{aligned}$$

Step 2. The signer computes

$$\begin{aligned}
&C_1 = S'_1 \circ A \circ T'_1,\\
&C_2 = S'_2 \circ A \circ T'_2,\\
&\cdots\cdots\\
&C_q = S'_q \circ A \circ T'_q;
\end{aligned}$$

Then q sets of u quadratic equations with n variables are obtained:

$$y_k^{(1)} = \sum_i \sum_j \gamma_{ijk}^{(1)} x_i^{(1)} x_j^{(1)} + \sum_i \mu_{ik}^{(1)} x_i^{(1)} + \delta_k^{(1)}, \quad \text{for } k = 1, \cdots, u;$$
$$\cdots\cdots$$
$$y_k^{(q)} = \sum_i \sum_j \gamma_{ijk}^{(q)} x_i^{(q)} x_j^{(q)} + \sum_i \mu_{ik}^{(q)} x_i^{(q)} + \delta_k^{(q)}, \quad \text{for } k = 1, \cdots, u;$$

Step 3. The signer computes the hash value $H = \mathcal{H}(m \| C_1 \| C_2 \| \ldots \| C_q)$, where $\|$ is the concatenation function. The signer computes

$$(S_i, T_i) = \begin{cases} (S_i', T_i'), & if\ H[i] == 0 \\ (S_i' \circ S^{-1}, T^{-1} \circ T_i'), & if\ H[i] == 1 \end{cases}, \text{ for } i = 1, \ldots, q,$$

where $H[i]$ stands for the i-th bit of H;

Step 4. Let $V = (H, (S_1, T_1), (S_2, T_2), \ldots, (S_q, T_q))$, and return V;

End

Definition 2. (IP Signature) *The* $(q+1)$*-tuples*

$$V = (H, (S_1, T_1), (S_2, T_2), \ldots, (S_q, T_q)) \tag{5}$$

is called the IP Signature *on message* m *by the signer using public key* (A, B).

Verification of IP Signature. After receiving the public key (A, B), the message m , and the IP signature V, any verifier can invoke the following algorithm **IPVerify** to check the validity of the signature.

Algorithm 2. IPVerify($m, V, (A, B)$ **)**
// to verify the validity of an IP signature on a message;

Input

m: the message to sign;

V: the IP signature on m, here $V = (H, (S_1, T_1), \ldots, (S_q, T_q))$;

(A, B): the public key of the signer;

Output

$true/false$: identifying whether the IP signature V is valid or not;

Begin

Step 1. The verifier computes q sets of u quadratic equations via

$$C_i' = \begin{cases} S_i \circ A \circ T_i, & if\ H[i] == 0 \\ S_i \circ B \circ T_i, & if\ H[i] == 1 \end{cases}, \text{ for } i = 1, \ldots, q;$$

Step 2. The verifier computes the hash value $H' = \mathcal{H}(m \| C_1' \| C_2' \| \ldots \| C_q')$;

Step 3. if $(H' == H)$, return $true$; otherwise, return $false$;

End

---

3 Proposed IP-Based Proxy Signature

3.1 Initialization

Suppose that Alice and Bob are users, and their universal unique identifiers are ID_A and ID_B respectively. Alice is the original signer, and Bob is the proxy signer. Alice's private key consists of (S_A, T_A) which is a pair of randomly-chose invertible affine transformations and in the forms of (3) and (4), and the corresponding public key is $(F_A, \bar{F}_A)$ satisfying

$$\bar{F}_A = S_A \circ F_A \circ T_A, \tag{6}$$

where F_A and $\bar{F}_A$ are two set of u quadratic polynomials in the forms of (1) and (2). Similarly, Bob's private key consists of (S_B, T_B) which is a pair of randomly-chose bijective affine transformations and in the forms of (3) and (4), and the corresponding public key is $(F_B, \bar{F}_B)$ that satisfies

$$\bar{F}_B = S_B \circ F_B \circ T_B. \tag{7}$$

where F_B and $\bar{F}_B$ are two set of u quadratic polynomials in the forms of (1) and (2).

The universal unique identifiers and the public keys of Alice and Bob, i.e., $(ID_A, (F_A, \bar{F}_A))$ and $(ID_B, (F_B, \bar{F}_B))$, are published to the public bulletin board.

3.2 Delegation and Proxy Key Generation

At this stage, a delegation token, which is called the "proxy" to represent the proxy signing power authorized to Bob by Alice, is computed by Alice and delivered to Bob. Then Bob can generate the proxy signing key by invoking its own private key and the proxy. The detailed steps are as follows.

Step 1. [Proxy Generation]

Alice randomly chooses two bijective affine transformations S and T in the forms of (3) and (4) respectively, then computes

$$S_\sigma = S \circ S_A^{-1}, \tag{8}$$

$$T_\sigma = T_A^{-1} \circ T, \tag{9}$$

and

$$\bar{F}'_A = S_\sigma \circ \bar{F}_A \circ T_\sigma. \tag{10}$$

The affines S and T should be kept secret by Alice.

Alice sends $(S_\sigma, T_\sigma, \bar{F}'_A)$ to Bob via an authenticated channel.

Step 2. [Generation of the Proxy Signing Key]

After receiving $(S_\sigma, T_\sigma, \bar{F}'_A)$, Bob computes

$$S_{\tilde{\sigma}} = S_\sigma \circ {S_B}^{-1}, \tag{11}$$

$$T_{\tilde{\sigma}} = {T_B}^{-1} \circ T_\sigma, \tag{12}$$

and

$$\bar{F}'_{AB} = S_B \circ \bar{F}_A \circ T_B. \tag{13}$$

Note that $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ is a private key for IP signature, and the corresponding public key is $(\bar{F}'_{AB}, \bar{F}'_A)$, which is proved by Lemma 1.

Bob sends $\bar{F}'_{AB}$ to Alice via an authenticated channel.

Step 3. [Publishing the Warrant]

Let $\omega = (ID_A, ID_B, t, (\bar{F}'_{AB}, \bar{F}'_A))$ be a warrant that Alice delegates its privilege to Bob to generate proxy signature on behalf of Alice, where t is the period of time that Alice authorizes its proxy signing power to Bob.

Alice invokes the algorithm **IPSign** to generate an IP signature on ω using (S_A, T_A) as private key and $(F_A, \bar{F}_A)$ as public key:

$$V_A = \text{IPSign}(\omega, (S_A, T_A), (F_A, \bar{F}_A)).$$

Alice publishes (ω, V_A) to the public bulletin board.

Now Bob can use $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ as the signing key for the proxy signature.

Lemma 1. *$(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ is an isomorphism from $\bar{F}'_{AB}$ to $\bar{F}'_A$, and $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ is the private key for IP signature corresponding to the public key $(\bar{F}'_{AB}, \bar{F}'_A)$.*

Proof. The Eq.(10) can be converted to:

$$\bar{F}'_A = S_\sigma \circ \bar{F}_A \circ T_\sigma = S_\sigma \circ S_B^{-1} \circ S_B \circ \bar{F}_A \circ T_B \circ T_B^{-1} \circ T_\sigma. \tag{14}$$

By applying (11), (12) and (13) to (14), we can get

$$\bar{F}'_A = S_{\tilde{\sigma}} \circ \bar{F}'_{AB} \circ T_{\tilde{\sigma}}, \tag{15}$$

which means that $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ is an isomorphism from $\bar{F}'_{AB}$ to $\bar{F}'_A$, therefore $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ is the private key for IP signature corresponding to the public key $(\bar{F}'_{AB}, \bar{F}'_A)$.

3.3 Generation of Proxy Signature

For any given message m, Bob can invoke the algorithm **IPSign** to generate an IP signature using $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ as private key and $(\bar{F}'_{AB}, \bar{F}'_A)$ as public key:

$$V_B = \text{IPSign}(m, (S_{\tilde{\sigma}}, T_{\tilde{\sigma}}), (\bar{F}'_{AB}, \bar{F}'_A)). \tag{16}$$

As a result, (V_B, ω) is defined to be the proxy signature on m by Bob on behalf of Alice.

3.4 Verification of Proxy Signature

After receiving the message m and the proxy signature (V_B, ω), any verifier can check the validity of the proxy signature by executing the following steps:

Step 1. Get Alice's public key $(F_A, \bar{F}_A)$ and the signature V_A on ω from the bulletin board by using ID_A as querying condition.

Step 2. Check the validity of ω by verifying the following conditions:

1) Invoke the the algorithm **IPVerify** to check the validity of the signature V_A on ω by :

$$\text{IPVerify}(\omega, V_A, (F_A, \bar{F}_A)) == true?$$

2) Check whether or not t described in ω is a valid time period.

If both of the above conditions hold true, proceed to the next step; otherwise, terminate the protocol.

Step 3. Verify the proxy signature by invoking the algorithm

$$\text{IPVerify}(m, V_B, (\bar{F}'_{AB}, \bar{F}'_A)) == true?$$

If the above condition holds true, then the proxy signature is valid; otherwise, it is invalid.

4 Security Analysis

4.1 Security of IP Problem

Some security models, for example random oracle model and standard model, can be adopted to formally prove the security of traditional cryptosystems based on integer factoring, discrete logarithm, and/or elliptic curve. However, these security models cannot be directly applied to Multivariate Public Key Cryptosystems (MPKC). A lot of researchers have paid intensive attention to solve the problem of provable security of MPKCs, which is to prove that a given MPKC is indeed secure with some reasonable theoretical assumptions. Even though there have been some works related to this area, for example [5–7, 17], there are still little essential progress in this topic. Therefore, the most common ways to analyze the security of MPKCs are to launch all existing effective attacks on the targeted MPKC to test its ability against these attacks.

Cryptosystems based on IP problem belong to a major category of MPKCs. Many researches have been devoting their efforts to solve the IP problem in an efficient way. The "To and Fro" technique proposed by Patarin, Goubin, and Courtois [14] is a significant approach to solve the IP problem, which assumes the ability to invert the polynomial systems, then has an exponential complexity. Moreover, the authors give an upper bound on the theoretical complexity of IP problem. But Faugère and Perret [7] pointed out that the proof in [14] is not complete. They gave an upper bound on the theoretical complexity of "IP-like" problems, and presented a new algorithm to solve IP problem when S and T are linear mappings. An improved algorithm proposed by Bouillaguet, Faugère,

Fouque, and Perret [4] integrates some new techniques, and claims to get the best result on the state of the art.

An important special case of IP is the IP problem with one secret (IP1S for short). Although most of the algorithms for IP can be applied to IP1S almost directly, several efficient algorithms are proposed to solve IP1S problem.

The algorithm to solve IP1S proposed by Geiselmann and Meier [10] conducts an exhaustive search to find the solutions of an algebraic system of equations. Later, Levy-dit-Vehel and Perret improved it by using the Gröbner basis computation [19]. Perret presented a new approach for solving IP1S using the Jacobian matrix [16], and the computational complexity of which is polynomial when the number of polynomials u is equal to the number of variables n, but is inefficient when u is much smaller than n.

We summarize, as far as we know, the best algorithms to solve the IP and IP1S problems in Table 1. We can observe that solving the IP problem is computationally hard if we choose the parameter properly, since the best algorithm for some categories of IP problem is still exponential.

Table 1. Best Algorithms to Solve IP and IP1S Problems

Problem	Subcase	Complexity
IP1S	degree = 2	$\mathcal{O}(n^6)$
	degree = 3, $u \ll n$, inhomogeneous	$\mathcal{O}(n^6)$
	degree = 3, $u \ll n$, homogeneous	$\mathcal{O}(n^6 \cdot m^{n/2})$
IP	degree = 2, $u = n$, inhomogeneous	Heuristic:$\mathcal{O}(n^3)$ / Rigorous:$\mathcal{O}(n^6)$
	degree = 2, $u = n$, homogeneous	$\mathcal{O}(n^{3.5} \cdot m^{n/2})$

Notation in Table 1:

u is the number of polynomials in A described in Eq.(1),
n is the number of variables,
and $m = ||K||$ is the cardinality of the finite field K.

4.2 Security of Our Scheme

We can know from (16) that the generation of the proxy signature is by invoking the ordinary IP signature and using the proxy signing key and public key. Therefore, as long as the procedure of proxy delegation is secure, our proxy signature scheme should share the same security level with the underlying IP signature scheme.

With the knowledge of the most efficient attacks on the IP problems, in order to strengthen the security of our scheme, we suggest that the parameters of our scheme should satisfy the following conditions:

1) the transformations S and T in Eq.(3) and (4) should be affine;

2) the polynomials in A and B in Eq.(1) and (2) should be homogeneous;

3)the number of polynomials u should be smaller than the number of variables n.

Taking our implementation in Section 5 as an example, if the parameters we choose are as follows: the finite field $K = GF(2^8), n = 18, u = 10$, and $m = ||K|| = 2^8$, then the attacking complexity should be greater than

$$\mathcal{O}(n^{3.5} \cdot m^{n/2}) = 18^{3.5} \cdot (2^8)^{18/2} = 2^{86.59}.$$

Usually, it is considered to be a computationally secure MPKC scheme if the attacking complexity is greater than 2^{80}. Therefore, our implementation and the parameters we choose should be a secure IP-based proxy signature scheme.

4.3 Properties of the Proposed Proxy Signature

Some properties of proxy signature are discussed in the section. It is assumed that all the secret or private information should be kept secret by its owners, including the private keys of Alice and Bob, the proxy signing key $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ and Alice's secret affines (S, T) in Section 3.2.

On Unforgeability of Ordinary Signature. The proxy signer Bob cannot derive the original signer Alice's private key (S_A, T_A) from the received message $(S_\sigma, T_\sigma, \bar{F}'_A)$, since the affines S and T are kept secret by Alice. Besides, anyone who gets $(S_\sigma, T_\sigma, \bar{F}'_A)$ by eavesdropping or other methods, for example disclosing the information by Bob intentionally or unintentionally, can not compute Alice's private key, either. Therefore it is difficult for attackers to get Alice's private key to forge its ordinary signature.

On Unforgeability of Proxy Signature. From (11) and (12) we learn that the signing key $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ for proxy signature is computed by invoking Bob's private key (S_B, T_B), which means that anyone cannot derive $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ without the knowledge of the proxy signer's private key. Therefore, only the proxy signer Bob can create a valid proxy signature on behalf of the original signer Alice.

On Secret-Key's Dependence. The signing key $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ for proxy signature is computed from Alice's private key (S_A, T_A) and Bob's private-key (S_B, T_B), which is shown by equations (8), (9), (11) and (12).

On Verifiability. From (10) and (13) we know that new public key $(\bar{F}'_{AB}, \bar{F}'_A)$ are computed only from the original public key $\bar{F}_A$, and the use of $\bar{F}_A$ is an evidence for the original signer's agreement.

On Distinguishability. On the one hand, the ordinary signature created by the original signer Alice is verified by using Alice's public key $(F_A, \bar{F}_A)$. On the other hand, the proxy signature consists of two parts: ordinary digital signature V_B and the warrant ω. The verification of V_B uses another public key $(\bar{F}'_{AB}, \bar{F}'_A)$ for proxy signature. Therefore, the use of different public keys distinguish ordinary signatures and proxy signatures.

Identifiability. The original signer can identify the proxy signer from $\bar{F}'_A$. Since no one except the original signer can generate with high probability a legal pair of (S_σ, T_σ) and $\bar{F}'_A$ for a target public key, only the person that has given the (S_σ, T_σ) is the signer capable to create the corresponding proxy signatures. In addition, the signature V_A on ω along with the identity information ID_B in ω, which is a part of the proxy signature, can also identify the proxy signer.

Therefore, an original signer can determine from a proxy signature the identity of the corresponding proxy signer.

Undeniability. Since the signing key $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ for proxy signature is computed by invoking Bob's private key (S_B, T_B), the proxy signer Bob cannot deny a proxy signature it has created, as in ordinary digital signatures.

Revocability. If the time period t in the warrant ω is expired, the delegated signing privilege is revoked automatically. Besides, the original signer can also broadcast a signed message to announce the invalidation of the warrant ω. Then the proxy signature generated by Bob hereafter will become invalid.

Strong Unforgeability. The original signer and third parties who are not designated as proxy signers cannot create a valid proxy signature, since the signing key $(S_{\tilde{\sigma}}, T_{\tilde{\sigma}})$ for proxy signature is computed by invoking the proxy signer Bob's private key (S_B, T_B).

Strong Identifiability. Anyone can determine the identity of the corresponding proxy signer from a proxy signature, since the warrant ω is a part of the proxy signature and contains the identity information of the corresponding proxy signer.

Strong Undeniability. A proxy signer cannot repudiate a proxy signature it created, as in ordinary digital signatures.

5 Implementation and Performance

Our proposed scheme is implemented in C/C++ programming language. The hardware and software of the machine to run our program is as follows: the CPU is Intel(R) Core(TM)2 Duo CPU E6550 @ 2.33GHZ, the memory is 1GB, the operating system is Windows XP, and the programming environment is Microsoft Visual Studio 2008.

The parameters described in Section 2 is assigned the values as follows: the finite field $K = GF(2^8), n = 18, u = 10$, and $q = 64$. As mentioned in Section 4.1, the security level of our scheme adopting these parameters can be up to $2^{86.59}$ and can reach the practical security requirement.

The complexity and running time of each procedure is shown in Table 2. We can observe that our implementation can efficiently generate a proxy signature in about 93 ms, and the verification of proxy signature takes about 266 ms.

Table 2. Time Complexity and Running Time of each Procedure

Procedure	Time Complexity	Running Time(ms)
Delegation computation by Alice	$\mathcal{O}(3u^3 + 3n^3 + 2un^3 + qun^3)$	110
Delegation computation by Bob	$\mathcal{O}(2u^3 + 2n^3 + 3un^3 + 2qun^3)$	203
Generate of proxy signature	$\mathcal{O}(qun^3)$	93
Verification of proxy signature	$\mathcal{O}(3qun^3)$	266

6 Conclusion

A novel proxy signature scheme based on the problem of Isomorphism of Polynomials (IP) is proposed, which is possible to be the first proxy signature scheme based on Multivariate Public Key Cryptography (MPKC). The main advantage of our scheme over other proxy schemes should be its post-quantum feature, since MPKC and IP cryptsystems can potentially resist the future quantum computing attacks. Through security discussion, our scheme can reach the same security level as the IP signature, which is a signature scheme based on IP problem. Some major properties of proxy signature schemes, such as strong unforgeability, strong identifiability, strong undeniability, secret-key's dependence, distinguishability, etc., are also owned by our scheme. The proposed scheme is implemented in C/C++ programming language. The measured performance shows that our implementation can run efficiently, and the parameters we choose can let the security level of the scheme up to $2^{86.59}$, which is considered to be computationally secure.

Acknowledgement. This paper is financially supported by the National Natural Science Foundation of China under Grant No. U1135004 and 61170080, and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2011), and Guangzhou Metropolitan Science and Technology Planning Project under grant No. 2011J4300028, and High-level Talents Project of Guangdong Institutions of Higher Education (2012), and the Fundamental Research Funds for the Central Universities under Grant No. 2009ZZ0035 and 2011ZG0015, and Guangdong Provincial Natural Science Foundation of under grant No. 9351064101000003.

The authors thank Ms. Li Yang and Mr. Guangdong Yang, who were the former students of the first author, for doing the implementation to verify the applicability of our scheme, and the discussion with whom also help improve some details of this paper.

References

1. Awasthi, A., Lal, S.: Proxy Blind Signature Scheme. Transaction on Cryptology 2(1), 5–11 (2005)
2. Bernstein, D.J., Buchmann, J., Dahmen, E.: Post Quantum Cryptography, Department of Computer Science, University of Illinois, Chicago. Springer, Heidelberg (2009)

3. Boldyreva, A., Palacio, A., Warinschi, B.: Secure Proxy Signature Schemes for Delegation of Signing Rights. Journal of Cryptology 25(1), 57–115 (2012)
4. Bouillaguet, C., Faugère, J., Fouque, P., Perret, L.: Differential Algorithms for the Isomorphism of Polynomials Problem (2009) (manuscript), http://eprint.iacr.org/2009/583.pdf
5. Bulygin, S., Petzoldt, A., Buchmann, J.: Towards Provable Security of the Unbalanced Oil and Vinegar Signature Scheme under Direct Attacks. In: Gong, G., Gupta, K.C. (eds.) INDOCRYPT 2010. LNCS, vol. 6498, pp. 17–32. Springer, Heidelberg (2010)
6. Dubois, V., Granboulan, L., Stern, J.: An Efficient Provable Distinguisher for HFE. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 156–167. Springer, Heidelberg (2006)
7. Faugère, J.-C., Perret, L.: Polynomial Equivalence Problems: Algorithmic and Theoretical Aspects. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 30–47. Springer, Heidelberg (2006)
8. Fuchsbauer, G., Pointcheval, D.: Anonymous Proxy Signatures. In: Ostrovsky, R., De Prisco, R., Visconti, I. (eds.) SCN 2008. LNCS, vol. 5229, pp. 201–217. Springer, Heidelberg (2008)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-completeness. W.H. Freeman (1979)
10. Geiselmann, W., Meier, W.: An Attack on the Isomorphisms of Polynomials Problem with One Secret. International Journal of Information Security 2, 59–64 (2003)
11. Kim, S., Park, S., Won, D.: Proxy Signatures, Revisited. In: Information and Communications Security, pp. 223–232. Springer, Heidelberg (1997)
12. Mambo, M., Usuda, K., Okamoto, E.: Proxy Signatures: Delegation of The Power to Sign Messages. IEICE Transactions on Fundamentals, E79-A(9), 1338-1353 (1996)
13. Mambo, M., Usuda, K., Okamoto, E.: Proxy Signatures for Delegating Signing Operation. In: Proceedings of the 3rd ACM Conference on Computer and Communications Security, pp. 48-57. ACM (1996)
14. Patarin, J., Goubin, L., Courtois, N.T.: Improved Algorithms for Isomorphisms of Polynomials. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 184–200. Springer, Heidelberg (1998)
15. Patarin, J.: Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 33–48. Springer, Heidelberg (1996)
16. Perret, L.: A Fast Cryptanalysis of the Isomorphism of Polynomials with One Secret Problem. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 354–370. Springer, Heidelberg (2005)
17. Sakumoto, K., Shirai, T., Hiwatari, H.: On Provable Security of UOV and HFE Signature Schemes against Chosen-Message Attack. In: Yang, B.-Y. (ed.) PQCrypto 2011. LNCS, vol. 7071, pp. 68–82. Springer, Heidelberg (2011)
18. Shor, P.W.: Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. SIAM J. Comput. 26(5), 1484–1509 (1997)
19. Levy-dit-Vehel, F., Perret, L.: Polynomial Equivalence Problems. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 235–251. Springer, Heidelberg (2003)
20. Zhang, K.: Threshold Proxy Signature Schemes. In: Okamoto, E. (ed.) ISW 1997. LNCS, vol. 1396, pp. 282–290. Springer, Heidelberg (1998)

Universal Designated Verifier Signcryption

Fei Tang, Changlu Lin, and Pinhui Ke

Key Laboratory of Network Security and Cryptology
Fujian Normal University, Fujian, 35007, China
tangfei127@163.com, cllin@fjnu.edu.cn

Abstract. Universal Designated Verifier Signature (UDVS) was introduced by Steinfeld et al. in Asiacrypt'03. UDVS allows a signature holder, who has a signature of a signer, to convince a designated verifier that he is in possession of a signer's signature, while the verifier cannot transfer such conviction to anyone else. In existing designs of UDVS, a secure channel is required between the signer and the signature holder for signature transmission. In this paper, we eliminate that requirement by combining the notions of UDVS and signcryption, and for the first time, propose the notion of *universal designated verifier signcryption* (UDVSC). We provide formal definitions and a concrete universal designated verifier signcryption scheme in the identity-based setting.

Keywords: Universal designated verifier, signcryption scheme, identity-based, random oracle.

1 Introduction

The concept of designated verifier signature (DVS) was proposed by Jakobsson et al. [11] in Eurocrypt'96. In DVS schemes, an entity called designated verifier can produce signatures which are indistinguishable from those generated by the signer. Thus, any third party cannot tell whether a signature was generated by the signer or by the designated verifier. In this way, a signature produced by the signer can only convince the designated verifier chosen by the signer. Due to this property, DVS has found very useful in many settings, such as call for tenders and digital vote, where the signer's privacy is a concern, and a number of designs have been proposed [10,11,18].

Motivated by the privacy concerns on disseminating signed digital certificates, Steinfeld et al. [18] proposed a new notion called *Universal Designated Verifier Signature* (UDVS). A UDVS scheme involves three parties: a signer, a signature holder and a (or multi-) designated verifier. The signer signs a message and sends the signature to the receiver who is then called signature holder. To prove to another entity that the signer has signed the message, the signature holder can send the signature to that entity. However, this also allows that entity to convince anyone else about the signer's endorsement on the message. Another more feasible way is that the signature holder converts the signer's signature to a UDVS which is designated to a verifier, such that only this designated verifier

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 126–134, 2012.

can believe the signer's endorsement on the message. Any other third parties will not believe it as the designated verifier can use his/her private key to create a valid UDVS, which is indistinguishable from the one produced by the signature holder. Like DVS, the designated verifier of UDVS can believe that the signer has signed the message, as he/she did not generate that UDVS. When the signature holder and the signature signer are the same user, a UDVS scheme will form a DVS scheme. Therefore, UDVS can be viewed as an application of general DVS.

In previous UDVS schemes, to prevent signature exposure, a secure channel between the signer and the receiver is necessary for signature delivery. In practical applications, generally, this is achieved using a secure encryption scheme. In this paper, we combine the notions of UDVS and signcryption and propose the notion of *Universal Designated Verifier Signcryption* (UDVSC) for the first time. The new notion preserves all desirable properties of UDVS and eliminates the need of secure channel for signature delivery. As shown in [23], a secure signcryption scheme provides both confidentiality and authentication in a more efficient way than signature add encryption, i.e., *Cost (Signcryption)* << *Cost (Signature) + Cost (Encryption)*. In this paper, we adopt Chen et al.'s signcryption scheme [7] to construct a concrete ID-based universal designated verifier signcryption scheme and prove its security in the random oracle model.

1.1 Related Works

The notion of UDVS was first proposed by Steinfeld et al. [18] in the Asiacrypt'03. Steinfeld et al. [19] then extended the standard Schnorr and RSA signature schemes to UDVS schemes. In 2005, Zhang et al. [22] proposed a short UDVS scheme based on BB signature scheme [2] in the standard model. Laguillaumie et al. [12] also proposed a UDVS scheme in the standard model, but their scheme is based on Waters' signature [21]. In Asiacrypt'05, Baek et al. [1] proposed two universal designated verifier signature proof (UDVSP) systems based on BLS signature scheme [4] and BB signature scheme [2], respectively. In their schemes, they adopted an interactive proof protocol mentioned in paper [5] and eliminated the register process of designated verifier's secret-public key pairs. Ng et al. [14] proposed a UDMVS scheme which allows the signature holder to efficiently produce UDVS for multi-verifier. In ICALP'06, Vergnaud [20] extended the pairing-based signatures to UDVS (or UDMVS) schemes. In 2008, Seo et al. [16] proposed a UDMVS scheme based on identity-based cryptosystem. Cao et al. [6] then presented an ID-based UDVS scheme in the standard model based on Waters' signature [21]. In 2009, Chen et al. [8] proposed an ID-based UDVSP system using interactive proof protocol, and Shahandashti et al. [17] described a generic construction for UDVS scheme from a large class of signature schemes.

In IAS'09, Li et al. [13] proposed an attack on Baek et al.'s schemes [1] and showed that their schemes do not satisfy the claimed purpose, i.e., the non-transferability. To be more specific, a malicious designated verifier in UDVSP schemes [1] can convert the interactive proof into a non-interactive signature which can convince anyone that the signer has endorsed the message. The same problem can also be found in Chen et al.'s [8] schemes.

1.2 Our Contributions

In this paper, we combine the notions of UDVS and signcryption schemes, and introduce the notion of *universal designated verifier signcryption* for the first time. We present a concrete design of universal designated verifier signcryption in the identity-based setting. The new scheme has the following properties:

- At the beginning, the signer signs a message and sends the signature to a designated receiver. Different from previous UDVS schemes, the signature can be transmitted via a public channel in our scheme.
- Upon receiving the signature, the designated receiver can efficiently transform the original signature and send the transformational signature to a designated verifier chosen by the receiver.
- Finally, the designated verifier can easily verify the validity of transformational signature but can not convince anyone else that the signer has endorsed the message.

1.3 Organization

In the next section, we review some preliminaries. Formal definitions and security models of the new ID-UDVSC scheme are defined in Section 3. The concrete ID-UDVSC scheme presented in Section 4. In Section 5, we only present the main security results and its proofs are given in the full version of the paper. We conclude in Section 6.

2 Preliminaries

2.1 Notations

Following notations will be used in this paper: "$a \leftarrow \mathsf{A}(x)$", the short left arrow denotes an algorithm A with input x and then output a; "$w := v$" denotes the assignment of a value v to w; for simplicity, id_s, id_{dr}, id_{dv} and PKG denote the signer, designated receiver, designated verifier and trusty private key generator, respectively; sk_i and pk_i denote the user i's secret and public key, respectively, where $i \in \{\mathsf{id}_s, \mathsf{id}_{dr}, \mathsf{id}_{dv}\}$.

2.2 Bilinear Mapping

Let G_1, G_2 and G_T be cyclic multiplicative groups with a same order p (it is a suitable prime such that computationally infeasible to solve the **Discrete-Logarithm** problem in G_1, G_2 and G_T); g_1 (g_2) is an arbitrary generator of G_1 (G_2). We say that a mapping $\hat{e} : G_1 \times G_2 \rightarrow G_T$ is an admissible bilinear mapping if it satisfies the following properties:

- *Bilinearity*: $\hat{e}(g_1^a, g_2^b) = \hat{e}(g_1, g_2)^{ab}$ for all $a, b \in Z_p$;
- *Non-degeneracy*: there exists $g_1^c \in G_1, g_2^d \in G_2$, where $c, d \in Z_p$, such that $\hat{e}(g_1^c, g_2^d) \neq 1_{G_T}$, ($1_{G_T}$ denotes G_T's identical element); and

– *Computability*: there exists an efficient algorithm to compute $\hat{e}(g_1^a, g_2^b)$ for all $a, b \in Z_p$.

We assume there is an isomorphism $\psi : G_2 \to G_1$ such that $\psi(g_2) = g_1$. In a specific case, we can set $G_2 = G_1$ and $g_2 = g_1 = g$. Please refer to [3] for more on the definition and design of bilinear mappings.

2.3 Bilinear Diffie-Hellman Problem (BDH)

Given a randomly chosen $g \in G_1$, as well as g^a, g^b, g^c (for unknown $a, b, c \in Z_p^*$), the BDH problem is to compute $\hat{e}(g, g)^{abc} \in G_T$. It is generally accepted that the BDH problem is computationally infeasible with a suitable bilinear mapping.

3 Formal Definitions and Security Models

3.1 Formal Definitions

Our ID-UDVSC scheme consists of the following seven algorithms: system setup, users' secret key extracting, signcrypting, unsigncrypting, transforming, transforming by the designated verifier, designated verifying. They are denoted by Setup, $\mathsf{Extract}$, $\mathsf{Signcrypt}$, $\mathsf{Unsigncrypt}$, $\mathsf{Transform}$, $\overline{\mathsf{Transform}}$ and $\mathsf{D\text{-}Verify}$, respectively. The detailed descriptions are as follows.

– Setup: A probabilistic algorithm, taken as input a security parameter k, outputs the system master key msk and public parameters $params$. That is: $(msk, params) \leftarrow \mathsf{Setup}(k)$.
– $\mathsf{Extract}$: A probabilistic algorithm, taken as input id, outputs user id's secret key sk_{id}. That is: $sk_{\mathsf{id}} \leftarrow \mathsf{Extract}_{msk}(\mathsf{id})$.
– $\mathsf{Signcrypt}$: A probabilistic algorithm, taken as input $(sk_s, \mathsf{id}_{dr}, m)$, outputs a ciphertext δ. That is: $\delta \leftarrow \mathsf{Sign}_{sk_s, \mathsf{id}_{dr}}(m)$.
– $\mathsf{Unsigncrypt}$: A deterministic algorithm, taken as input (sk_{dr}, δ), outputs the signature σ that is sealed in the δ if it is valid. Otherwise output "Rej". That is: "σ/Rej" $\leftarrow \mathsf{Verify}_{sk_{dr}, \mathsf{id}_s}(\sigma)$,
– $\mathsf{Transform}$: A probabilistic algorithm, taken as input $(\mathsf{id}_{dv}, \sigma)$, outputs a transformational signature $\tilde{\sigma}$. That is: $\tilde{\sigma} \leftarrow \mathsf{Transform}_{\mathsf{id}_{dv}}(\sigma)$.
– $\overline{\mathsf{Transform}}$: A probabilistic algorithm, taken as input $(sk_{dv}, \mathsf{id}_s, m')$, outputs a transformational signature $\tilde{\sigma}'$ on behalf of the designated verifier. That is: $\tilde{\sigma}' \leftarrow \overline{\mathsf{Transform}}_{sk_{dv}, \mathsf{id}_s}(m')$.
– $\mathsf{D\text{-}Verify}$: A deterministic algorithm, taken as input $(sk_{dv}, \tilde{\sigma})$, outputs "$Acc$" if it is a valid transformational signature. Otherwise output "Rej". That is: "Acc/Rej" $\leftarrow \mathsf{D\text{-}Verify}_{sk_{dv}}(\tilde{\sigma})$.

3.2 Security Models

Confidentiality. To ensure only the designated receiver can obtain the signature, the confidentiality of the signature of the ID-UDVSC scheme is necessary. Consider the following experiment $\mathsf{Exp}_{\mathcal{A}}^{ind-idudvsc-cca}(k)$ which is played by an $\mathsf{IND\text{-}IDUDVSC\text{-}CCA}$ adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- **Setup:** The challenger $\mathcal{C}$ generates the public parameters $params$, and sends it to the $\mathcal{A}$.
- **Phase 1:** $\mathcal{A}$ is able to take some adaptive queries to $\mathcal{C}$ and $\mathcal{C}$ must respond with corresponding answers.
  - *Extract queries*: $\mathcal{A}$ takes id as input to this oracle, and $\mathcal{C}$ returns sk_{id} to $\mathcal{A}$.
  - *Signcrypt queries*: $\mathcal{A}$ takes $(m, \mathsf{id}_s, \mathsf{id}_{dr})$ as inputs to this oracle, and $\mathcal{C}$ returns a corresponding ciphertext δ to $\mathcal{A}$ as answer.
  - *Unsigncrypt queries*: $\mathcal{A}$ takes $(\delta, \mathsf{id}_{dr})$ as inputs to this oracle, and $\mathcal{C}$ returns σ if it is a valid signature, else "Rej", to $\mathcal{A}$ as answer.
  - *Transform queries*: $\mathcal{A}$ takes $(\sigma, \mathsf{id}_{dv})$ as input to this oracle, and $\mathcal{C}$ returns a transformational signature $\tilde{\sigma}$ to $\mathcal{A}$ as answer.
  - $\overline{Transform}$ *queries*: $\mathcal{A}$ takes $(m', \mathsf{id}_s, \mathsf{id}_{dv})$ as input to this oracle, and $\mathcal{C}$ returns a transformational signature $\tilde{\sigma}'$ to $\mathcal{A}$ as answer.
  - *D-Verify queries*: $\mathcal{A}$ takes $(\tilde{\sigma}, \mathsf{id}_{dv})$ as input to this oracle, and $\mathcal{C}$ returns Acc if it is valid. Otherwise, return "Rej".
- **Challenge:** After finishing the execution of phase 1, $\mathcal{A}$ outputs two messages $\{m_0, m_1\}$ and two identities $\{\mathsf{id}_s^*, \mathsf{id}_{dr}^*\}$ which are wish to be challenged. $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$ and runs the signcrypting algorithm to generate a ciphertext δ^*, then sends it to $\mathcal{A}$.
- **Phase 2:** After receiving δ^*, $\mathcal{A}$ can make more queries as in Phase 1 but:
  1. id_{dr}^* is never taken as input during *Extract queries*.
  2. $(\delta^*, \mathsf{id}_{dr}^*)$ is never taken as input during *Unsigncrypt queries*.
- **Guess:** Finally, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$. We say that $\mathcal{A}$ wins the above experiment if $b' = b$. The advantage of $\mathcal{A}$ is defined as $\mathbf{Adv}_{\mathcal{A}}^{ind-idudvsc-cca}(k) = |\mathbf{Pr}[b' = b] - \frac{1}{2}|$.

Definition 1. *We say that an ID-based universal designated verifier signcryption scheme is* IND-IDUDVSC-CCA *secure if no polynomial-bounded adversary has a non-negligible advantage* **Adv** *to win the above experiment* $\mathsf{Exp}_{\mathcal{A}}^{ind-idudvsc-cca}(k)$.

Unforgeability. We adapt Huang et al.'s [9] security definition, *existential unforgeability of the transformational signature against adaptive chosen message attacks*, to the new ID-UDVSC scheme. Consider the following experiment $\mathsf{Exp}_{\mathcal{A}}^{eu-idudvsc-cma}(k)$ which is played by an EU-IDUDVSC-CMA adversary $\mathcal{A}$ and a challenger $\mathcal{C}$.

- **Setup:** As in the experiment $\mathsf{Exp}_{\mathcal{A}}^{ind-idudvsc-cca}(k)$.
- **Queries:** As in the phase 1 in the experiment $\mathsf{Exp}_{\mathcal{A}}^{ind-idudvsc-cca}(k)$.
- **Output:** We say that $\mathcal{A}$ wins this experiment if $\mathcal{A}$ outputs a transformational signature $\tilde{\sigma^*}$ with identities $\mathsf{id}_s^*, \mathsf{id}_{dv}^*$ such that:
  1. "Acc" $\leftarrow \mathsf{D\text{-}Verify}_{sk_{dv}^*}(\tilde{\sigma^*})$.
  2. $\{\mathsf{id}_s^*, \mathsf{id}_{dv}^*\}$ are not taken as inputs during the *Extract queries*.
  3. (m^*, id_s^*) is not taken as input during the *Signcrypt queries*, where the m^* is the message corresponding to the forgery.
  4. $(m^*, \mathsf{id}_s^*, \mathsf{id}_{dv}^*)$ is not taken as input during the $\overline{Transform}$ *queries*.

Definition 2. *We say that an ID-based universal designated verifier signcryption scheme is EU-IDUDVSC-CMA secure if no polynomial-bounded adversary has a non-negligible advantage* **Adv** *to win the above experiment* $\mathsf{Exp}_{\mathcal{A}}^{eu-idudvsc-cma}(k)$.

Non-transferability. The purpose of the non-transferability is to protect the signature receiver's privacy, which means that the designated verifier cannot convince any third party that the known signer has signed on the message m. We also adapt Huang et al.'s [9] security definition, *non-transferability against adaptive chosen message attack*, to the new ID-UDVSC scheme. Consider the following experiment $\mathsf{Exp}_{\mathcal{D}}^{nt-idudvsc-cma}(k)$ which is played by a challenger $\mathcal{C}$ and a distinguisher $\mathcal{D}$.

- **Setup:** As in the experiment $\mathsf{Exp}_{\mathcal{D}}^{eu-idudvsc-cma}(k)$ swapping $\mathcal{A}$ with the distinguisher $\mathcal{D}$.
- **Phase 1:** As in the experiment $\mathsf{Exp}_{\mathcal{D}}^{eu-idudvsc-cma}(k)$ swapping $\mathcal{A}$ with the distinguisher $\mathcal{D}$.
- **Challenge:** After finishing the execution of phase 1, $\mathcal{D}$ submits the information $(m^*, \mathsf{id}_s, \mathsf{id}_{dv})$ to $\mathcal{C}$ as the challenge. Then, the challenger $\mathcal{C}$ randomly chooses a bit $b \in \{0, 1\}$.
  1. If $b = 1$, $\mathcal{C}$ runs the $\mathsf{Transform}$ algorithm and returns a transformational signature $\tilde{\sigma}^*$ to $\mathcal{D}$.
  2. If $b = 0$, $\mathcal{C}$ runs the $\overline{\mathsf{Transform}}$ algorithm and returns a transformational signature $\tilde{\sigma}^{*'}$ to $\mathcal{D}$.
- **Phase 2:** After receiving the σ^* or $\tilde{\sigma}^{*'}$, $\mathcal{D}$ can make more queries as in phase.
- **Guess:** Finally, $\mathcal{D}$ outputs his guess bit $b' \in \{0, 1\}$. We say that $\mathcal{D}$ wins the this experiment if $b' = b$. The advantage of $\mathcal{D}$ is defined as $\mathbf{Adv}_{\mathcal{D}}^{nt-idudvsc-cma}(k) = |\mathbf{Pr}[b' = b] - \frac{1}{2}|$.

Definition 3. *We say that an ID-based universal designated verifier signcryption scheme is NT-IDUDVSC-CMA secure if no polynomial-bounded adversary has a non-negligible advantage* **Adv** *to win the above experiment* $\mathsf{Exp}_{\mathcal{D}}^{nt-idudvsc-cma}(k)$.

Remark: In this experiment, the distinguisher $\mathcal{D}$ wants to obtain some information via making queries to the above oracles, and thus distinguish the transformational signature $\tilde{\sigma}'$ (generated by the designated verifier) from $\tilde{\sigma}$ (generated by the designated receiver). Our definition requires that no one can tell the difference between those two kinds of signatures.

4 Proposed ID-UDVSC Scheme

- Setup: Run the setup algorithm to generate the system parameters and master key:
  1. $(G_1, G_T, \hat{e}, p, g)$ are the elements of the bilinear mapping that described in Section 2.2.

2. Choose random $x \in Z_p^*$ as the master key and compute $y = g^x \in G_1$ as master public key.
3. $H_0 : \{0,1\}^{k_0} \to G_1, H_1 : \{0,1\}^{k_1+n} \to Z_p^*$ and $H_2 : G_T \to \{0,1\}^{k_0+k_1+n}$ are three collision-resistant hash functions, where k_0, k_1, n denote the sizes of users' identity, element of G_1, message m, respectively.
4. Finally, this algorithm makes the system parameters $params :=< G_1, G_T, \hat{e}, p, g, y, H_0, H_1, H_2 >$ publicly, and keeps the master key $msk := x$ secretly.

– Extract: PKG computes user's secret key: $sk_{\mathsf{id}} = H_0(\mathsf{id})^x \in G_1$, and distributes it to the corresponding user via a secure channel.
– Signcrypt: Let $m \in \{0,1\}^n$ to be signcrypted by id_s, who chooses random $r \in Z_p$ and computes:

$$R = H_0(\mathsf{id}_s)^r, \qquad h = H_1(R||m),$$
$$S = sk_s^{(r+h)}, \qquad T = \hat{e}(sk_s^r, H_0(\mathsf{id}_{dr})),$$
$$\alpha = H_2(T) \oplus (S||\mathsf{id}_s||m).$$

The ciphertext is $\delta := (R, \alpha)$.
– Unsigncrypt: After receiving δ, id_{dr} recovers the original signature and verifies its validity as follows:

$$T = \hat{e}(R, sk_{dr}), \qquad (S||\mathsf{id}_s||m) = \alpha \oplus H_2(T),$$
$$h = H_1(R||m), \qquad \hat{e}(S, g) \stackrel{?}{=} \hat{e}(y, R \cdot H_0(\mathsf{id}_s)^h).$$

Output $\sigma := (m, R, S, \mathsf{id}_s)$ if the verification equation holds. Otherwise output "Rej".
– Transform: if id_{dr} wants to prove the signature σ to id_{dv}, he computes $V = \hat{e}(S, H_0(\mathsf{id}_{dv}))$, then sends the transformational signature $\tilde{\sigma} := (V, R, m, \mathsf{id}_s)$ to id_{dv}.
– Transform: id_{dv} also can generate a transformational signature $\tilde{\sigma}' := (V', R', m', \mathsf{id}_s)$ for an arbitrary message m' on the behalf the receiver: choose random $R' \in G_1, m' \in \{0,1\}^n$, and compute

$$h = H_1(R'||m'), \qquad V' = \hat{e}(sk_{dv}, R' \cdot H_0(\mathsf{id}_s)^h).$$

– D-Verify: Finally, id_{dv} verifies $\tilde{\sigma}$'s validity via checking the verification equation $V \stackrel{?}{=} \hat{e}(sk_{dv}, R \cdot H_0(\mathsf{id}_s)^h)$, where $h = H_1(R||m)$, holds or not, output "Acc" if it holds. Otherwise output "Rej".

5 Security Results

In this section, we only present the main security results of the proposed ID-UDVSC scheme, the detailed proofs are showed in the full version of this paper.

Theorem 1. (IND-IDUDVSC-CCA) *The proposed ID-based universal designated verifier signcryption scheme is indistinguishable against adaptive chosen ciphertext attack if the hardness of BDH problem holds.*

Theorem 2. (EU-IDUDVSC-CMA) *The proposed ID-based universal designated verifier signcryption scheme is existentially unforgeable against adaptive chosen message attack if the hardness of BDH problem holds.*

Theorem 3. (NT-IDUDVSC-CMA) *The proposed ID-based universal designated verifier signcryption scheme is non-transferable against adaptive chosen message attack.*

6 Conclusion

In this paper, we proposed the notion of universal designated verifier signcryption and presented an identity-based universal designated verifier signcryption scheme. In the new notion, the privacy of signature holder can be guaranteed even in a public transmission channel, and thus it greatly improves the practical applicability of normal universal designated verifier signatures. The security proofs for our proposed ID-UDVS scheme are given in the random oracle model.

Acknowledgment. This research is supported in part by the National Natural Science Foundations of China under Grant No. 61103247/61102093, the Natural Science Foundation of Fujian Province under Grant No. 2011J05147, and the Foundation for Excellent Young Teachers of Fujian Normal University (No.fjsdjk2012049).

References

1. Baek, J., Safavi-Naini, R., Susilo, W.: Universal Designated Verifier Signature Proof (or How to Efficiently Prove Knowledge of a Signature). In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 644–661. Springer, Heidelberg (2005)
2. Boneh, D., Boyen, X.: Short Signatures Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
3. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
4. Boneh, D., Lynn, B., Shacham, H.: Short Signatures from the Weil Pairing. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 514–582. Springer, Heidelberg (2001)
5. Cramer, R., Damgård, I.B., Schoenmakers, B.: Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
6. Cao, F., Cao, Z.: An Identity Based Universal Designated Verifier Signature Scheme Secure in the Standard Model. Journal of Systems and Software 82, 643–649 (2009)
7. Chen, L., Malone-Lee, J.: Improved Identity-Based Signcryption. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 362–379. Springer, Heidelberg (2005)
8. Chen, X., Chen, G., Zhang, F., Wei, B., Mu, Y.: Identity-Based Universal Designated Verifier Signature Proof System. International Journal of Network Security 8(1), 52–58 (2009)

9. Huang, X., Susilo, W., Mu, Y., Wu, W.: Secure Universal Designated Verifier Signature without Random Oracles. Int. J. Inf. Secur. 7, 171–183 (2008)
10. Huang, X., Susilo, W., Mu, Y., Zhang, F.T.: Short (Identity-Based) Strong Designated Verifier Signature Schemes. In: Chen, K., Deng, R., Lai, X., Zhou, J. (eds.) ISPEC 2006. LNCS, vol. 3903, pp. 214–225. Springer, Heidelberg (2006)
11. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
12. Laguillaumie, F., Libert, B., Quisquater, J.-J.: Universal Designated Verifier Signatures Without Random Oracles or Non-black Box Assumptions. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 63–77. Springer, Heidelberg (2006)
13. Li, Y., Pang, L., Wang, Y.: Attacks on a Universal Designated Verifier Signature Scheme. In: Proceedings of the Fifth International Conference on Information Assurance and Security, vol. 1, pp. 27–30 (2009)
14. Ng, C.Y., Susilo, W., Mu, Y.: Universal Designated Multi Verifier Signature Schemes. In: Proceedings of the 2005 11th International Conference on Parallel and Distributed Systems, pp. 305–309. IEEE (2005)
15. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology 13, 361–369 (2000)
16. Seo, S.H., Hwang, J.Y., Choi, K.Y., Lee, D.H.: Identity-based Universal Designated Multi-Verifiers Signature Schemes. Computer Standards and Interfaces 30, 288–295 (2008)
17. Shahandashti, S.F., Safavi-Naini, R.: Genric Constructions for Universal Designated-Verifier Signatures and Identity-based Signatures from Standard Signatures. IET Information Security 3(4), 152–176 (2009)
18. Steinfeld, R., Bull, L., Wang, H., Pieprzyk, J.: Universal Designated-Verifier Signatures. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 523–542. Springer, Heidelberg (2003)
19. Steinfeld, R., Wang, H., Pieprzyk, J.: Efficient Extension of Standard Schnorr/RSA Signatures into Universal Designated-Verifier Signatures. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 86–100. Springer, Heidelberg (2004)
20. Vergnaud, D.: New Extensions of Pairing-Based Signatures into Universal Designated Verifier Signatures. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 58–69. Springer, Heidelberg (2006)
21. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
22. Zhang, R., Furukawa, J., Imai, H.: Short Signature and Universal Designated Verifier Signature Without Random Oracles. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 483–498. Springer, Heidelberg (2005)
23. Zheng, Y.: Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost(Signature) + Cost(Encryption). In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 165–179. Springer, Heidelberg (1997)

A Bird's Eye View on the I2P Anonymous File-Sharing Environment

Juan Pablo Timpanaro[1], Isabelle Chrisment[2], and Olivier Festor[1]

[1] INRIA Nancy-Grand Est, France
[2] LORIA - ESIAL, Université de Lorraine
{juanpablo.timpanaro,olivier.festor}@inria.fr,
isabelle.chrisment@loria.fr

Abstract. Anonymous communications have been gaining more and more interest from Internet users as privacy and anonymity problems have emerged. Among anonymous enabled services, anonymous file-sharing is one of the most active one and is increasingly growing. Large scale monitoring on these systems allows us to grasp how they behave, which type of data is shared among users, the overall behaviour in the system. But *does large scale monitoring jeopardize the system anonymity?*

In this work we present the first large scale monitoring architecture and experiments on the I2P network, a low-latency message-oriented anonymous network. We characterize the file-sharing environment within I2P, and evaluate if this monitoring affects the anonymity provided by the network.

We show that most activities within the network are file-sharing oriented, along with anonymous web-hosting. We assess the wide geographical location of nodes and network popularity. We also demonstrate that group-based profiling is feasible on this particular network.

Keywords: Large scale monitoring, I2P, Security risks, Anonymous file-sharing.

1 Introduction

Anonymous communications have been acquiring more and more interest since the past decade, either for fighting against any type of censorship, passive attacks (third-parties sniffing, traffic analysis, user profiling) or for malicious purposes (copyrighted material downloads). Within anonymous communications, anonymous file-sharing is one of the most active fields and is increasingly growing, partially due to the onrush of negative news on public file-sharing, including legal actions by governmental institutions, law-enforcement agencies and movie maker associations to major file-sharing communities, and partially because of the rising concern of both privacy and anonymity concepts within the Internet.

Large scale monitoring on file-sharing communities provides a wide view of the network[1][2][3] and allows us to answer the following questions: *what kind of content is mainly distributed? How many users does the network have? How many files does the network hold? Which users are downloading a given content?*

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 135–148, 2012.

However, large scale monitoring on anonymous environments has not been widely investigated, and is mainly focused on the Tor network [4][5]. Anonymous systems often imply a decoupling between the identity of users and their activities within the system, hardening the monitoring. Among anonymous file-sharing systems, this means that it becomes very challenging to successfully link together a specific user with a specific download.

The I2P[6] network is a low-latency message-oriented anonymous network designed as a network layer, in which any two users can communicate among themselves in completely anonymous manner. This network has a full range of available applications: anonymous web-browsing, chatting, file-sharing, web-hosting, e-mail and blogging among others. Except for anonymous web-browsing that necessarily requires an out-proxy to the normal Internet, the rest of the mentioned applications interact between each other within the network boundaries.

In this paper, we provide an efficient method to monitor the I2P anonymous file-sharing community. We perform an analysis of the I2P network based on intense monitoring, which to the best of our knowledge is the first large scale monitoring effort of the network, answering the following questions: *is it possible to properly characterize the I2P anonymous file-sharing system? Does the anonymity provided by the I2P network get compromised by large scale monitoring? Does large scale monitoring introduce new security risks?*. Our goal is to properly characterize the I2P file-sharing environment, and assess if a large scale monitoring activity jeopardizes the anonymity of the network.

This paper is organized as follows: Section 2 introduces the I2P network and its main components and features, including used encryption techniques, peer profiling and its distributed database. Section 3 describes our monitoring architecture. We detail how we use I2P's distributed database to collect data and how we use this data to characterize the file-sharing side of the network. An experiment is presented in section 4, in which we deploy our monitoring architecture over several days. Section 5 points out previous and current work on anonymous file-sharing and large scale monitoring. Finally, Section 6 concludes this work.

2 The I2P Network

The I2P network allows anonymous communications between two parties through an abstraction layer, which uncouples the association between the user and its identity. Basically, an application using I2P will not longer be reachable through an IP, but through a *location independent identifier*. The following sub-sections address the I2P network and its features.

2.1 Overview

The network is formed by a group of *routers*, which is the software that allows any application to communicate through I2P. Applications running on top of it will have a *destination* associated, which receives incoming connections from third parties. The secret lies in which destination is associated to which router and

not in the fact that a user is running an instance of the router. This decoupling between the router and the destination provides a certain degree of anonymity.

I2P uses a variation of the well-known onion routing approach[7], in which a message is routed from its originator to the final endpoint through several intermediate nodes using layered encryption. This variation is called *Garlic routing*, in which several messages along with their delivery instructions can be encapsulated into a single message and encrypted with the receiver's key. The integrity required for garlic messages are achieved through an hybrid *ElGamal/AES+Session Tags* symmetric-asymmetric approach.

The path through a selected list of nodes is called a *tunnel* and it is one of the key concepts in I2P.

2.2 Tunnels

Every tunnel is unidirectional, and is formed by the gateway (entry point), a set of participants (intermediate nodes) and an endpoint (exit point). Two types of tunnels exist. *Inbound* tunnels allow a user to receive data, and *outbound* tunnels to send data. A fully bidirectional communication between two users will involve four tunnels, one inbound and one outbound for each user.

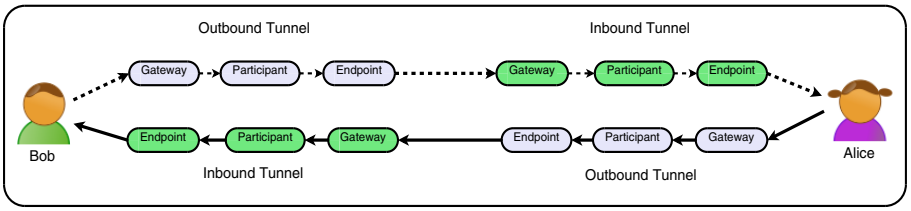


Fig. 1. Simple tunnel-oriented communication in I2P

Figure 1 illustrates a communication between Alice and Bob. Alice sends a message through her outbound tunnel, targeting one of Bob's inbound tunnel gateway. Once the message reaches the *gateway*, which is the entry point for Bob's tunnel, it is forwarded all the way through Bob's router. Alice does not have knowledge about Bob's inbound tunnel, but only about the entry point: Bob might have as well a tunnel composed with 1 or 100 intermediate nodes, but Alice ignores this, leading to the earlier mentioned *decoupling*.

2.3 Profiling Algorithm

Tunnels are created every 10 minutes, and then dropped. This feature hardens a traffic analysis attack, since a 10 minutes time window is rather small to acquire sufficient knowledge of the network. Hence, every 10 minutes, a user needs to select new nodes for its tunnels, which are selected among all the routers in the network.

I2P leans on a constantly local profiling of all seen routers, so as to characterize every peer regarding its performance, latency, and availability. A four-tier scheme

is used to classify routers: fast and high capacity, high capacity, not failing, and failing. Tunnels participants are randomly chosen from the fast and high capacity tier, and order through out the tunnel according to the *XOR* distance from a random key value. This *XOR* ordering prevents information leakage in predecessor and harvesting attacks.

Profiles are based on lookups in the network database, how often messages through remote routers fail, how many new routers are these remote routers able to introduce to us, for example. All kind of indirected behaviour is recorded and used for profiling, to the contrary of claimed performance from routers. No published performance information is used in local profiling, so as to avoid attacks based on announcing highly performant routers.

2.4 I2P netDB

The *netDB* is another key concept within the I2P network. It is a distributed hash table based on the Kademlia[8] protocol, used to store and share network metadata. However, on the contrary to the Kademlia protocol, not every peer in the I2P network forms part of the DHT, but only those fast and performant I2P users, the so called *floodfill* peers.

Any I2P user can become a floodfill node if two conditions are met: the number of estimated floodfills in the network is less than 250 nodes, and the user's I2P router has more than 128 KB/s of available bandwidth assigned.

There are two types of network metadata, *leasesets* and *routerinfos*. A leaseset provides information about a specific destination, like a web server, a BitTorrent client, an e-mail server, etc. A routerinfo provides information about a specific router and how to contact it, including the router identity (keys and a certificate), the address where to contact it (IP and port), several text options and a signature.

This distributed database contains an extra security feature, to harden a localized *Sybil attack*. The key used to index a record in the netDB is computed as `HASH(ID + date)`, in which the *ID* is the record ID, and *date* is the current date. A record ID remains fixed as long as the record is conserved, however the record indexing (routing) key changes every day.

This produces that at midnight, all indexing keys will be changed and therefore re-published in other locations in the DHT. Even though some queries might fail around midnight, this approach avoids an attacker to launch a simply localized Sybil attack, since the attacker will have to re-compute its Sybils IDs using a key-to-key dictionary. A deeper view of this network database is out of the scope of this document, however a further insight of the netDB and the flooding mechanism can be found in the official I2P website[6].

3 I2P Monitoring Approach

Our primary goal is to monitor the I2P network by qualifying file-sharing within the network, and study if this monitoring presents any kind of anonymity risk.

As earlier mentioned, I2P bases its anonymity on decoupling the identity of a user (provided by its routerinfo) from the application it is running (provided by its leaseset). Therefore, our challenge is to determine whether a given leaseset is running any known I2P file-sharing application, even if we can not link a particular user with a specific file-sharing application. This section introduces our monitoring architecture, and how it is implemented.

3.1 Exploting I2P netDB

As described previously, the netDB stores every routerinfo and every published leaseset. We aim to retrieve from the netDB as many leasesets as possible, for further analysis. Higher the number of leasesets retrieved and analysed, higher will be the characterization of the network.

We exploit the mechanism for becoming a floodfill, mentioned in Section 3.5, by placing a set of modified floodfill nodes in the netDB, which behave exactly as normal floodfills from the point of view of network operations (storage of routerinfos and leasesets), but perform a deeper analysis of these leasesets.

Number of Floodfills. It is important to consider the netDB coverage of our architecture, and to determine how many monitoring nodes are needed to have a full (or good enough) network coverage.

Let N be the number of floodfills and X the replica factor, we consider the minimum number of floodfill monitor nodes as:

$$\texttt{nb\_monitors} = \lceil\ \texttt{N / X}\ \rceil\texttt{, N= \#floodfills , X= replica factor} \quad (1)$$

The replica factor indicates in how many netDB participants a given record (either a routerinfo or a leaseset record) is stored, for improving fault tolerance against participants going off-line. The I2P netDB defines a replica factor of 8, which means that any record is replicated in 8 participants for storing. If we consider that there are currently around 200 floodfill nodes in the I2P netDB and the replica factor of 8, we need 25 **perfectly distributed** monitor floodfill nodes out of the 200 nodes, to have a complete coverage.

Distribution of Floodfills. In equation 1 we assume that the monitoring nodes are perfectly distributed over the DHT space, which is not always the case. Even if the SHA2 hashing function used in the netDB assures a fairly well distribution of the nodes through out the DHT-space, it might happen that our monitoring nodes are *all together*.

Figure 2 displays two examples of how the same set of monitoring floodfill nodes might be positioned in the nedDB: the upper one shows all the monitoring nodes grouped in one side, whilst the lower one shows a more dispersed distribution.

It is clear that the *position* of the nodes plays a significant role in our architecture, and makes the result of equation 1 the lower bound of the floodfill

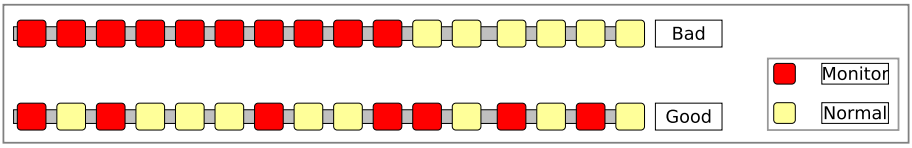


Fig. 2. DHT nodes distributions

monitoring nodes required. If the *positioning* of the nodes is not dispersed, we require further additional monitoring nodes.

Choosing the placement of our floodfill nodes will assure us a perfect distribution, however as mentioned in section 3.5, we would need a key-to-key dictionary to perform this. Nonetheless, we can verify whether our monitoring nodes are well distributed or not, by considering their position in the DHT-space along with the *optimal distance*. The optimal distance represents the distance between any two monitoring nodes for a correct distribution in the DHT-space.

It is important to consider that figure 2 represents the DHT-space as a linear space for an easy visualization. However, the I2P netDb is a Kademlia-based implementation, and therefore the concept of distance is based on the XOR metric, rather than in a linear distance.

Equation 2 presents the optimal distance in terms of bits for a group of monitor nodes, given a 2^{256} DHT-space, such as the I2P netDB. This states how many bits of difference must exist between the floodfills routing keys, to achieve a fairly dispersed distribution.

$$\texttt{optimal\_distance} = \log_2(2^{256}/MN), \texttt{ MN= Monitors nodes} \quad (2)$$

With 2^4 monitoring nodes for example, there must be a difference of $\log(2^{256}/2^4) = 252$ bits for a perfect distribution and an optimal coverage of the DHT-space.

3.2 File-Sharing Application Analysis

I2PSnark, IMule and I2Phex are the three main I2P available file-sharing applications. We analyse each of them in the following manner:

I2PSnark. Firstly, we establish a connection with the leaseset (similar to a TCP socket-like connection). Then we send a first message, a well-formed BitTorrent message, requesting a non-existing torrent ID. If that given leaseset is actually running an I2PSnark client, it will immediately close the connection. Secondly, we re-open the connection and send a malformed BitTorrent message. In the only case this response timeouts, then we can conclude the given leaseset is running an I2PSnark client.

IMule. An IMule client needs two different leasesets, one for its TCP connections (for file-transfer) and one for UDP messages (for indexation). We send an

eDonkey `hello_packet` in case for a TCP-like leaseset, and a KAD `hello_request` in case for an UDP-like leaseset. If any of the messages have a response, then we conclude this leaseset is running an IMule client.

I2Phex. A `GNUTELLA CONNECT` message is sent for testing a leaseset against an I2Phex Gnutella client. If there is a Gnutella-protocol response, then the leaseset is running a I2Phex client.

3.3 I2P Monitoring Architecture

Figure 3 presents our complete monitoring architecture. This architecture has a set of distributed probes (monitor floodfills) dispersed around the I2P netDB for data collection (`routerinfo` or `leaseset` records), while a group of databases are used to determine the geographical location of these records. All retrieved data is stored in a central database, for a further correlation analysis and a final display in our statistics website.

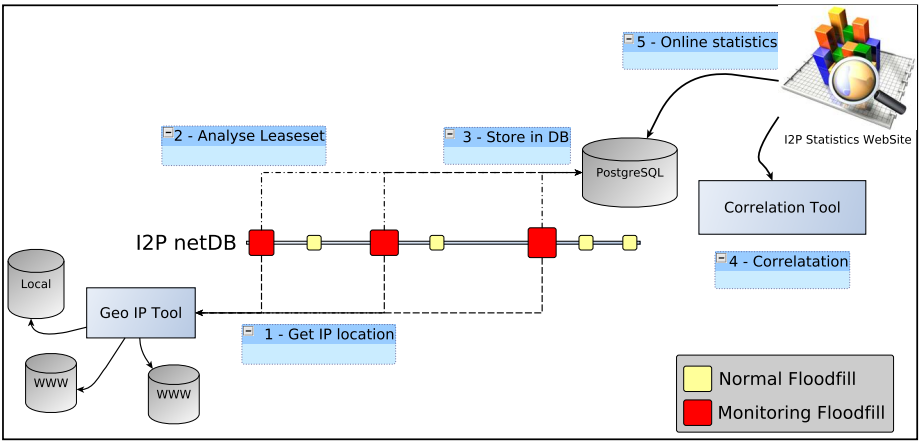


Fig. 3. Monitoring architecture

When a monitor floodfill receives a `routerinfo store message`, it looks up the estimated location of its IP address and stores the record in the central database. If a `leaseset store message` is received, the monitor floodfill runs the analysis for file-sharing applications. A periodically analysis is computed, including correlation between top countries and most used file-sharing applications, top cities, top file-sharing application, correlation between file-sharing applications, number of retrieved leasesets and routerinfos. All the results obtained are shown in a website, along with the state of the monitoring architecture.

On the one hand, our architecture is completely flexible: due to the autonomous nature of the monitoring floodfill nodes, the total number of these nodes can be increase at any time, increasing the amount of network metadata retrieved, therefore increasing the accuracy of the analysis.

On the other hand, the implementation of our monitoring floodfill nodes is freely available, along with its source code, hence any user willing to contribute to the analysis of the network can download a monitoring floodfill node and deploy it.

4 Experiments

In this section, we present our experiments on monitoring the I2P network for file-sharing applications.

4.1 Experiment Set-Up

We monitored the network for 90 hours, from 2012-06-07 15:00:00 (UTC+2) to 2012-06-11 09:00:00 (UTC+2). We contemplated a weekend in our measurements, being these days usually more suitable for file-sharing.

Based on the estimated number of 220 floodfill nodes in the I2P network[1] during our experiment, and in the equations presented in section 3.1, we needed a total of $\lceil\ 220\ /\ 8\ \rceil = 27.5$ floodfill monitor nodes for a perfect coverage, with a distance in terms of bits of $\log\left(2^{256}/27.5\right) = 251.67$ bits.

The PlanetLab test-bed was employed for this particular experiment, and due to technical limitations, only 20 monitor floodfills were deployed. By the time of the experiment, all of these floodfill nodes had an average previous uptime of 24 hours, so as to assure a good integration with the network.

Due to the particular indexation of records in the netDB, which includes the current date in the routing key calculation, the monitor floodfill nodes shift from one location to another one every day. We take our first day of experimentation, and calculates our monitor nodes distribution within the DHT-space.

A distance of 252 bits was computed between all of our floodfill nodes, which indicates that our floodfill nodes are fairly well distributed through the DHT-space. Taking into account this pseudo-perfect distribution, we can estimate that with 20 floodfill out of 27.5 we cover 72% of the network the first day of the experiment. Being that the monitor nodes move in the DHT-space, this estimation need to be done in a daily-basis.

It is important to consider that because the pseudo-perfect distribution of our nodes, we can cover 72% of the network. However, in a completely random distribution of monitor floodfill nodes within the netDB, the network coverage will drop. A further complete analysis can be found at `www.i2pstats.loria.fr`.

4.2 File-Sharing Applications within I2P

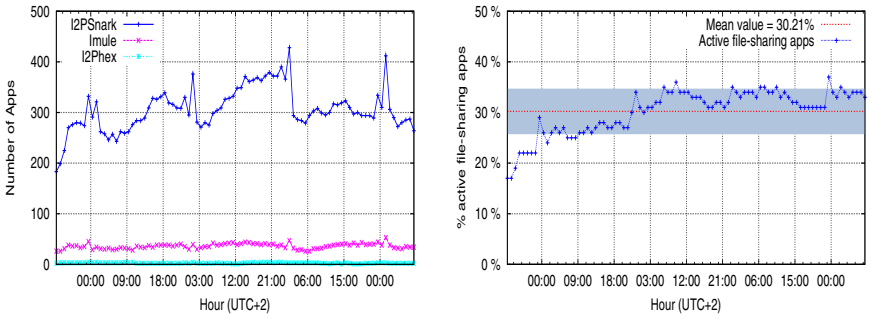
Figure 4(a) displays the individual number of active file-sharing applications during the course of our analysis, for I2PSnark, IMule and I2Phex clients.

I2PSnark clients usage highly exceed the rest of the measured applications, in which I2Phex presents an almost zero usage during the experiment, with as few as 11 clients detected.

[1] `http://stats.i2p.to/`

Figure 4(b) presents the file-sharing usage when compared to the total amount of analysed applications, in which it can be observed that during the weekend (right side of the graphic) the file-sharing usage increases up-to 38% of the total network usage, mainly guided by the I2PSnark clients, and it decreases during week days, which is an usual file-sharing characteristic. We compute an average usage of 30.21%, with a standard deviation of 4.38%.

4 well-defined peaks appear in the chart, all around midnight. This increase does not represent an increase in the real I2PSnark users base, but rather an exact view of how the netDB is implemented: at midnight every routing key changes, which means that every record (*routerinfo* or *leasest*) is re-located in another DHT location. Our floodfill monitor nodes receive these new storing requests, which increase the total number. However, previous stored records are not longer kept within our floodfill monitor nodes, therefore after the peaks at midnight, the measurement returns to normal.



(a) I2PSnark / IMule / I2Phex (b) File-sharing usage

Fig. 4. File-sharing applications within I2P

Why is it that I2PSnark prevail over IMule or I2Phex clients? I2PSnark is a BitTorrent-based client, which is fairly popular among file-sharing applications, and it is additionally built-in within the I2P software, which makes it rather easy to be used. Furthermore, the low number of IMule clients measured suggests a low number of sources (peers having the entire content) for IMule files, which leads to new clients choosing I2PSnark rather than IMule for file-sharing: less sources is translated into slow download rates or even no download at all.

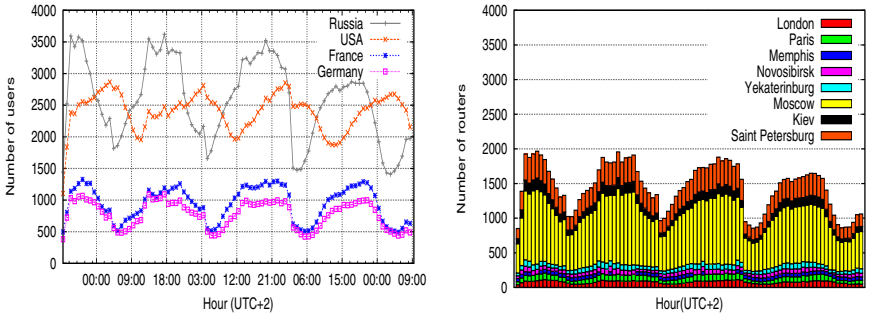
4.3 Country-Based Analysis

In this section we analyse users location, to determine which countries and cities contribute the most to the I2P network. End-users geographical location is represented by the geographical location of their routers within the I2P network.

Figure 5(a) presents the most active countries within the I2P network: Russia, The United States, France and Germany.

Russia and The United States present both a participation over 1500 routers at any point in time (with few exceptions). However their activity is inversely

proportional through time in this graphic, probably because Moscow and Saint-Petersbourg (most seen Russian cities), and Memphis (most seen U.S. city) have a time difference of 10 hours, therefore the U.S. curve is out of phase with the *European* time-zone, which does not imply that U.S. participants behave different that European ones.



(a) Top 4 active countries (b) Top active cities

Fig. 5. Country/City distribution

Figure 5(b) displays the most seen cities, in which Russian cities predominate over the rest of the cities. Moscow seems to be the most active Russian city detected, with more than 50% of participation in most cases. In both cases, country-based and city-based, Russia contributes the most to the I2P usage, with an average of 40% of the total participants in the network.

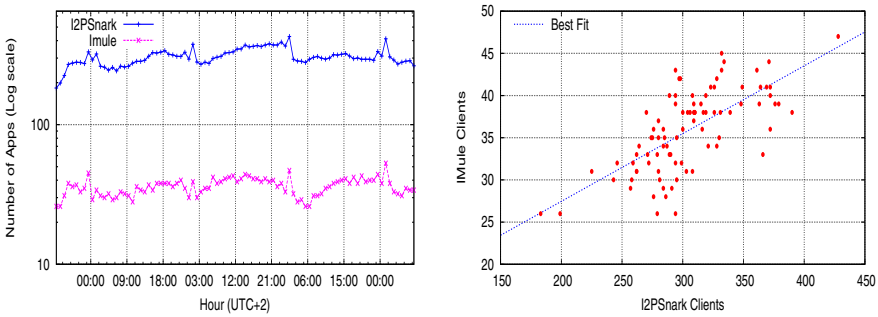
During the whole experiment, we detected 140 countries in total, and considering the 4 top countries, we identified 500 different Russian cities, 2130 different U.S. cities, 1670 different French cities, and 1654 different German cities. It is clear that I2P usage is not confined to a narrow group of countries or cities, but rather distributed all over the world.

4.4 Correlation between File-Sharing Applications

In this section we try to determinate if users of different file-sharing applications behave the same way: *does an increase of I2PSnark clients necessary imply an increase of IMule clients?*. We do not consider I2Phex clients, due to its low number of measured clients in the network.

Figure 6(a) displays a detailed view (log scale on the Y-axis) of figure 4(a), in which an increase on IMule clients can be deducted when an increase of I2PSnark clients occurs. Figure 6(b) presents a scatter plot with I2PSnark and IMule clients usage, revealing a positive 0.7106 correlation value among the usage of these two file-sharing clients. In this case, file-sharers behave the same way regardless which file-sharing applications they are using.

In our previous work[9] we show that eMule/aMule clients stay connected for longer periods of time, on the contrary to BitTorrent users. This behaviour is not seen in this case, most likely due to the fact that for I2PSnark or IMule



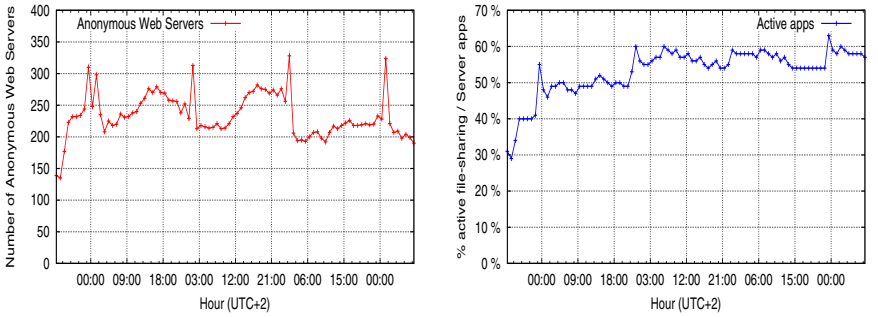
(a) Detailed view (b) I2PSnark/IMule Pearson-correlation

Fig. 6. File-sharing applications correlation

applications to run, an I2P user needs to have a running I2P router. Being that I2P users are more active at night, file-sharing applications will also be active during night time, independently of the type of the applications, either I2PSnark or IMule.

4.5 Non File-Sharing Applications

The I2P network allows anonymous web sites, called *eepSites*, in which any user can host an anonymous web server, and through a DNS-like service, I2P users can resolve domain names such as `tracker2.postman.i2p` (a major I2P tracker). Due to this built-in feature, we additionally measured anonymous web servers within I2P, by sending a `GET` message to a leaseset and processing the answer.



(a) Anonymous Web Servers (b) Aggregated usage

Fig. 7. Anonymous web servers usage

Figure 7(a) displays the number of active anonymous web servers measured during our experiments, with an average value of 240 online servers at any time and an average of 22.9% out of the total usage of the network. In this case, there

is no noticeable increase of servers during the weekend, which indicates that anonymous servers are quite stable within the network. In this measurement we can observed the same behaviour as mention in Section 4.2, in which at midnight new record storing messages arrive.

Both file-sharing and anonymous hosting reach to almost 65% of the total usage of the network, as seen in figure 7(b). The increase in the right part of the graphic is driven by file-sharing users, more than anonymous web servers.

5 Related Work

Among anonymous low-latency networks, the *Tor* network is probably the one that has been receiving more academic attention.

Chaabane et al.[4] conduct a traffic analysis in Tor by quantifying BitTorrent and HTTP protocols, and successfully detecting that Tor exit nodes are used as 1-hop SOCKS proxies rather than in a full-length onion tunnel.

Le Blond et al.[10] present a monitoring study in Tor, in which they successfully trace TCP streams within the network, and unveil BitTorrent users using the Tor network.

McCoy et al.[5] present a monitoring approach which analyses the application layer of outgoing traffic to determinate the protocol distribution in the Tor network, based on a Tor exit node. They detect that most of the connections through Tor are interactive http traffic, and that a very few of them are for BitTorrent traffic. However, these few BitTorrent connections (3.33%) consume a disproportionally amount of bandwidth (40.20% of the total measured).

Loesing et al.[11] introduce a measurement of sensitive data on the Tor network, such as country of connections of users and exiting traffic by port. Furthermore, Loesing [12] measures the *trends* of the Tor network from directory information.

Nevertheless, the Tor network has a central *directory server* and hence the monitoring approaches based on this central component can not be applied in the I2P network, which does not include any kind of centralized directory.

In our previous work[13] we take the first step through monitoring the application layer of the I2P network by means of a centralized architecture and measure a single file-sharing application along with I2P's hidden services.

Adrian Crenshaw[14] exhibits a mechanism to identify I2P's hidden services, called *eppSites*. He successfully links an anonymous website to its real IP address, taking advantage of the lack of control in the application layer, and misconfiguration of web servers over I2P. Crenshaw's approach includes crawling his local list of known participants along with the list of known *eepSites*, whereas our approach focus on the complete netDB.

Herrmann et al.[15] conducted an attack on the I2P network, so as to determining the identity of peers hosting *eepSites*, which can be considered a monitoring technique for anonymous websites. They proposed a three-step attack, in which an adversary with modest resources can identify an I2P user as the host of an *eepSite*. Their focus on a particular victim, rather than on the entire network.

Large scale monitoring on further popular anonymous low-latency networks, such as JAP, Freenet or GNUnet, has not been performed or even proposed in the literature.

6 Conclusion

We have designed and successfully implemented and deployed the first large scale monitoring architecture for the I2P network, mainly focused on anonymous file-sharing. We are able to provide deeper insights about the behaviour of the network, providing us with correlation values among users locations and file-sharing applications usage, which clearly states an anonymity warning: group-based identification for file-sharing applications is possible within the network, even if single-user identification is not.

We have measured that file-sharing within I2P obtains an average of 30%, peaking 35% during weekends, in which the I2PSnark client is the most used over IMule and I2Phex. There appears also to be a correlation among Russian users and file-sharing applications even in a 90-hours analysis. An ongoing and longer monthly analysis confirms this correlation.

In addition, we measured 22.9% of anonymous web servers hosted within I2P, which along with file-sharing users, add up to the 65% of the total network performing these two activities.

A real-time analysis of the network can be found on our I2P statistics web site, `www.i2pstats.loria.fr`, which presents the top countries, top cities, amount of file-sharing applications, amount of online users, correlation among file-sharing applications and countries, etc. We additionally include a raw access to our database, so as to researchers can query and create their own requests.

Future work targets the design of an improved technique for optimal placement of our monitor floodfill nodes, in which every monitor node chooses its placement in the DHT-space, achieving a perfect DHT distribution, thus increasing the total coverage of the network, whilst reducing the required number of monitoring nodes.

References

1. Siganos, G., Pujol, J.M., Rodriguez, P.: Monitoring the Bittorrent Monitors: A Bird's Eye View. In: Moon, S.B., Teixeira, R., Uhlig, S. (eds.) PAM 2009. LNCS, vol. 5448, pp. 175–184. Springer, Heidelberg (2009)
2. Steiner, M., En-Najjary, T., Biersack, E.W.: A global view of Kad. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement, IMC 2007, San Diego, California, USA. ACM (October 2007)
3. Banerjee, A., Faloutsos, M., Bhuyan, L.N.: The P2P War: Someone Is Monitoring Your Activities! In: Akyildiz, I.F., Sivakumar, R., Ekici, E., de Oliveira, J.C., McNair, J. (eds.) NETWORKING 2007. LNCS, vol. 4479, pp. 1096–1107. Springer, Heidelberg (2007)

4. Chaabane, A., Manils, P., Ali Kaafar, M.: Digging into Anonymous Traffic: A Deep Analysis of the Tor Anonymizing Network. In: Proceedings of the 2010 4th International Conference on Network and System Security. IEEE Computer Society, Washington, DC (2010)
5. McCoy, D., Bauer, K., Grunwald, D., Kohno, T., Sicker, D.C.: Shining Light in Dark Places: Understanding the Tor Network. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 63–76. Springer, Heidelberg (2008)
6. I2P. The I2P network, `http://www.i2p2.de/`
7. Goldschlag, D., Reed, M., Syverson, P.: Hiding Routing Information. In: Anderson, R. (ed.) IH 1996. LNCS, vol. 1174, pp. 137–150. Springer, Heidelberg (1996)
8. Maymounkov, P., Mazières, D.: Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 53–65. Springer, Heidelberg (2002)
9. Timpanaro, J.P., Cholez, T., Chrisment, I., Festor, O.: When KAD meets BitTorrent - Building a Stronger P2P Network. In: Proceedings of the 8th International Workshop on Hot Topics in Peer-to-Peer Systems, HotP2P 2011, Anchorage, Alaska, USA. IEEE Computer Society (May 2011)
10. Blond, S.L., Manils, P., Chaabane, A., Kaafar, M.A., Castelluccia, C., Legout, A., Dabbous, W.: One bad apple spoils the bunch: exploiting P2P applications to trace and profile Tor users. In: Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats, LEET 2011, Boston, MA. USENIX Association (March 2011)
11. Loesing, K., Murdoch, S.J., Dingledine, R.: A Case Study on Measuring Statistical Data in the Tor Anonymity Network. In: Sion, R., Curtmola, R., Dietrich, S., Kiayias, A., Miret, J.M., Sako, K., Sebé, F. (eds.) FC 2010 Workshops. LNCS, vol. 6054, pp. 203–215. Springer, Heidelberg (2010)
12. Loesing, K.: Measuring the Tor network from public directory information. In: Proceedings of the 4th Hot Topics in Privacy Enhancing Technologies, HotPETS 2011, Seattle, WA, USA. Springer (August 2009)
13. Timpanaro, J.P., Chrisment, I., Festor, O.: I2P's Usage Characterization. In: Pescapè, A., Salgarelli, L., Dimitropoulos, X. (eds.) TMA 2012. LNCS, vol. 7189, pp. 48–51. Springer, Heidelberg (2012)
14. Crenshaw, A.: Darknets and hidden servers: Identifying the true IP/network identity of I2P service hosts. In: Black Hat 2011, Washington, DC, USA (March 2011)
15. Herrmann, M., Grothoff, C.: Privacy-Implications of Performance-Based Peer Selection by Onion-Routers: A Real-World Case Study Using I2P. In: Fischer-Hübner, S., Hopper, N. (eds.) PETS 2011. LNCS, vol. 6794, pp. 155–174. Springer, Heidelberg (2011)

A Clustering-Based Approach for Personalized Privacy Preserving Publication of Moving Object Trajectory Data

Samaneh Mahdavifar[1], Mahdi Abadi[2],
Mohsen Kahani[1], and Hassan Mahdikhani[3]

[1] Department of Computer Engineering, Ferdowsi University of Mashhad, Iran
samaneh.mahdavifar@stu.um.ac.ir, kahani@um.ac.ir
[2] Department of Computer Engineering, Tarbiat Modares University, Tehran, Iran
abadi@modares.ac.ir
[3] School of Computer Engineering, University of Science & Technology, Tehran, Iran
h_mahdikhani@comp.iust.ac.ir

Abstract. With the growing prevalence of location-aware devices, the amount of trajectories generated by moving objects has been dramatically increased, resulting in various novel data mining applications. Since trajectories may contain sensitive information about their moving objects, so they ought to be anonymized before making them accessible to the public. Many existing approaches for trajectory anonymization consider the same privacy level for all moving objects, whereas different moving objects may have different privacy requirements. In this paper, we propose a novel greedy clustering-based approach for anonymizing trajectory data in which the privacy requirements of moving objects are not necessarily the same. We first assign a privacy level to each trajectory based on the privacy requirement of its moving object. We then partition trajectories into a set of fixed-radius clusters based on the EDR distance. Each cluster is created such that its size is proportional to the maximum privacy level of trajectories within it. We finally anonymize trajectories of each cluster using a novel matching point algorithm. The experimental results show that our approach can achieve a satisfactory trade-off between space distortion and re-identification probability of trajectory data, which is proportional to the privacy requirement of each moving object.

Keywords: privacy preservation, trajectory data, moving object, greedy clustering, space distortion, re-identification probability.

1 Introduction

In recent years, location-aware devices, such as Radio-Frequency Identification (RFID) tags, Global Positioning Systems (GPS), mobile phones, and point of sale terminals facilitate location-based services. Consequently, spatio-temporal locations of moving objects daily collected by service providers, are stored in

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 149–165, 2012.

the form of trajectories, making the objects easier to be tracked. In other words, trajectories are spatio-temporal traces of moving objects collected over months or years and contain precious information for various data mining techniques. Moving objects themselves typically refer to concrete objects (*e.g.*, humans, vehicles, animals, and goods), or even abstract concepts (*e.g.*, spreading diseases). In order to perform data mining tasks effectively, it is essential to have high-quality moving object trajectory data. However, trajectory data potentially contain particular information about their owners and disclosing such information may reveal their religious preferences, life styles, social customs, and sensitive personal information. Therefore, trajectory data cannot be published unless they are properly anonymized. Although extensive research has been conducted in trajectory data mining so far [1–5] resulting in many important real-life applications, such as traffic planning, intelligent transportation, location-based advertising, and homeland security, few works have been done on the realm of their privacy preservation.

The first method that comes to the mind is to remove identifying information, such as name and Social Security Number (SSN) from trajectories. Nevertheless, this naïve method does not ensure the privacy of moving objects even for simple relational data and is vulnerable to linking attacks. Because the published data can be joined with publicly available data through a set of common attributes, called quasi-identifiers, leading the sensitive information to be revealed. A recent study estimated that 87% of the population of the United States can be uniquely identified using the seemingly inoffensive attributes including, gender, date of birth, and 5-digit zip code [6]. Furthermore, the more the number of common attributes, the more the figure increases.

Spatio-temporal attributes are very powerful quasi-identifiers which can be used to link trajectory data to many other types of data. Imagine the trajectory of an employee that daily commutes to and from a specific location. Even if the identifiers in the trajectory are removed, it can be easily inferred that the starting point is a home, and the ending point is an office. Joining this information with some phone directories, an adversary can easily link the trajectory with the employee's identity.

The solution to protect de-identified data against linking attacks is anonymization [7, 8]. k-anonymity was initially proposed in the literature to anonymize relational data and preserve the privacy of individuals while revealing useful information. In a k-anonymized table, the records are anonymized using techniques such as generalization or suppression so that a given record cannot be distinguished from at least $(k-1)$ other records. Although it has been proven that finding an optimal solution to the k-anonymity problem is *NP*-hard, it remains a standard model to prevent privacy breaches in relational data. Nevertheless, in case of trajectory data, the natural complexity of spatio-temporal trajectories and the dependence of consecutive points in a trajectory make the problem of anonymization even harder.

In this paper, we propose a greedy clustering-based approach for anonymizing trajectory data, which meets different privacy requirements of different moving

objects. We first assign a privacy level to each trajectory based on the privacy requirement of its moving object. We then partition the trajectories into a set of fixed-radius clusters. To create each cluster, a single trajectory is randomly selected as the cluster centroid. The nearest trajectories to the cluster centroid with the EDR distance [9] less than a fixed threshold are next added into the cluster until its size is proportional to the maximum privacy level of trajectories within it. We finally anonymize trajectories of each cluster using a novel matching point algorithm.

The rest of the paper is organized as follows: Section 2 reviews related work. Section 3 presents a clustering-based approach for the anonymization of moving object trajectory data. Section 4 reports experimental results and finally Section 5 draws some conclusions.

2 Related Work

Few papers have tackled the problem of privacy preserving publication of moving object trajectory data so far [10–15]. Bonchi [16] has divided the existing methods for anonymization of moving object trajectory data into two different classes: methods based on quasi-identifier and methods based on clustering and perturbation.

2.1 Methods Based on Quasi-Identifier

This class of methods adopts some concepts of quasi-identifiers in anonymization of moving object trajectory data [10, 11]. Terrovitis *et al.* [10] consider trajectories as being simple sequences of addresses, corresponding to the places, and propose a method that iteratively suppresses selected locations from the original trajectories until a privacy constraint is met.

Unlike in microdata, it is very likely that various moving objects have different quasi-identifier attributes. Consequently, the anonymization groups of trajectories may not be disjoint and it is not possible to simply adopt the classical k-anonymity notions for trajectory data. To solve this problem, Yarovoy *et al.* [11] propose a notion of k-anonymity by defining an attack graph associated with the original data and its distorted version. They then present two different approaches, namely extreme-union and symmetric-anonymization, to build anonymization groups that provably satisfy the proposed k-anonymity requirement, as well as yield low information loss.

2.2 Methods Based on Clustering and Perturbation

The problem of defining a well-structured set of spatio-temporal quasi-identifiers is a challenging and probably out of the question task. Thus, this class of methods addresses the problem of anonymizing moving object trajectory data without considering any concept of quasi-identifiers, applying the concept of k-anonymity

[12–15]. Nergiz *et al.* [12] first extend the notion of k-anonymity and propose a heuristic method for trajectory anonymization. They then present a randomization based reconstruction algorithm for publishing anonymized trajectory data.

Abul *et al.* [13] propose a novel concept of k-anonymity based on space translation, called (k, δ)-anonymity, which makes use of the inherent uncertainty of locations in order to reduce the amount of distortion needed to anonymize trajectory data. They first recall their previous method, namely $\mathcal{NWA}$ [14], to produce clusters of trajectories, each having size in the interval $[k, 2k-1]$. Each cluster is perturbed by means of the minimum spatial translation needed to push all the trajectories within a common cylindrical volume of radius $\delta/2$. The main drawback of $\mathcal{NWA}$ is the fact that it is highly dependent on the Euclidean distance in clustering of trajectories. The Euclidean distance is not able to detect similar trajectories with local shifts and can only be defined between trajectories having exactly the same length. To solve this problem, they develop a clustering method based on the EDR distance, namely $\mathcal{W}4\mathcal{M}$, which has the important feature of being time-tolerant.

Monreale *et al.* [15] propose a method for the anonymization of trajectory data that combines spatial generalization and k-anonymity. They first construct a suitable tessellation of the geographical area into sub-areas that depends on the input trajectory data and apply a spatial generalization on it. They then transform the generalized trajectory data to ensure that it satisfies the notion of k-anonymity.

The main drawback of all the aforementioned methods is that they treat the privacy of different moving objects homogeneously. In other words, they consider the same privacy level for all moving objects. Moreover, they do not take the sequentiality of moving points into account while finding linked moving points between two trajectories.

3 Clustering-Based Approach

In this section, we propose a greedy clustering-based approach to anonymize moving object trajectory data in which the privacy requirements of moving objects are considered to be different. Before describing the details of the approach, we introduce some notations and basic definitions.

3.1 Notations and Basic Definitions

Given a set of moving objects O and a dataset of trajectories TR, the function ρ is defined as $\rho : TR \rightarrow O$ which maps each trajectory $tr_i \in TR$ into its moving object $\rho(tr_i) \in O$. For any moving object $o_j \in O$, the set of all its trajectories is denoted by $\tau(o_j)$:

$$\tau(o_j) = \{tr_i \in TR \mid \rho(tr_i) = o_j\} \quad . \tag{1}$$

Each trajectory $tr_i \in TR$ is represented as a sequence of spatio-temporal moving points in three-dimensional space:

$$tr_i = \langle p_i^1, \ldots, p_i^m \rangle \quad , \tag{2}$$

where $m \geq 2$ is the number of moving points in tr_i. A moving point $p_i^k \in tr_i$ is denoted by a triple (t_i^k, x_i^k, y_i^k), with (x_i^k, y_i^k) representing the position of tr_i at time t_i^k.

Definition 1 (Trajectory length). *The length of a given trajectory $tr_i \in TR$ is defined as the number of moving points it contains and is denoted by $|tr_i|$.*

Definition 2 (Trajectory tail). *The tail of the trajectory tr_i, denoted by $\theta(tr_i)$, is defined as all moving points in it except the leading one:*

$$\theta(tr_i) = \langle p_i^2, \ldots, p_i^m \rangle \ . \tag{3}$$

In this paper, we assume that different moving objects may have different privacy requirements. Therefore, we associate a privacy level with each moving object to represent its privacy requirement. Let $L = \{l_1, \ldots, l_{max}\}$ be a totally ordered set of privacy levels such that for all r, s if $r \leq s$ then $l_r \leq l_s$. We define $\ell : O \rightarrow L$ to be a total function that maps each moving object $o_j \in O$ to its privacy level $\ell(o_j)$. Hence, a privacy level $\ell(\rho(tr_i))$ is assigned to each trajectory $tr_i \in TR$.

Definition 3 (Trajectory privacy level). *The privacy level for a given trajectory is defined as the minimum number of other trajectories in the trajectory data from which it cannot be distinguished.*

Definition 4 (Linked and unlinked moving points). *Given a moving point $p_i^k \in tr_i$, the moving point $p_j^{k'} \in tr_j$ is called a linked moving point of p_i^k, if and only if*

$$|t_i^k - t_j^{k'}| \leq \delta_t, \sqrt{(x_i^k - x_j^{k'})^2 + (y_i^k - y_j^{k'})^2} \leq \delta_d \ , \tag{4}$$

where δ_t and δ_d are temporal and spatial thresholds, respectively. The set of linked moving points of p_i^k is denoted by $\phi(p_i^k)$.

Definition 5 (Distorted moving point). *Given a moving point $p_i^k \in tr_i$ and a spatial threshold δ_d, the moving point $p_i^{k'}$ is called a distorted moving point of p_i^k with respect to δ_d, denoted by $\zeta(p_i^k, \delta_d)$, if and only if*

$$|x_i^{k'} - x_i^k| = r, |y_i^{k'} - y_i^k| > \sqrt{\delta_d^2 - r^2} \ , \tag{5}$$

where r is a random value in the range $[0, \delta_d]$.

Definition 6 (Sub-trajectory). *Let $tr_i = \langle p_i^1, \ldots, p_i^m \rangle$ be a trajectory. A trajectory $tr_j = \langle p_j^1, \ldots, p_j^q \rangle$ is called a sub-trajectory of tr_i, denoted by $tr_j \sqsubseteq tr_i$, if there exist integers $1 \leq k_1 < \cdots < k_q \leq m$ such that*

$$p_j^1 \in \phi(p_i^{k_1}), p_j^2 \in \phi(p_i^{k_2}), \ldots, p_j^q \in \phi(p_i^{k_q}) \ . \tag{6}$$

As it is mentioned earlier, the Euclidean distance is not an appropriate distance function to measure the similarity between trajectories, due to its poor performance at the presence of local shifts and unequal trajectory lengths. In this

paper, we use Normalized Trajectory Distance (NTD) as a distance function for trajectories:

$$ntd(tr_i, tr_j) = \frac{edr(tr_i, tr_j)}{\max\{|tr_i|, |tr_j|\}} \ , \tag{7}$$

where $\max\{|tr_i|, |tr_j|\}$ is the maximum length of tr_i and tr_j. $edr(tr_i, tr_j)$ is the EDR distance [9] between tr_i and tr_j, recursively defined as

$$edr(tr_i, tr_j) = \begin{cases} |tr_i| & |tr_j| = 0 \ , \\ |tr_j| & |tr_i| = 0 \ , \\ \min\{edr(\theta(tr_i), \theta(tr_j)) + \xi(p_i^1, p_j^1), & \\ \quad edr(tr_i, \theta(tr_j)) + 1, & \text{otherwise} \ , \\ \quad edr(\theta(tr_i), tr_j) + 1\} & \end{cases} \tag{8}$$

where $\xi(p_i^1, p_j^1) = 0$ if $p_j^1 \in \phi(p_i^1)$ and $\xi(p_i^1, p_j^1) = 1$ otherwise.

3.2 Adversary's Background Knowledge

Let TR^* be the anonymized trajectory data. An adversary who gains access to TR^* may attempt to identify the trajectory tr_i of a target moving object $\rho(tr_i)$. We assume the adversary knows at most σ moving points that $\rho(tr_i)$ has previously visited. Such background knowledge about $\rho(tr_i)$ is denoted by ξ_i:

$$\xi_i = \langle p_i^1, \ldots, p_i^z \rangle \ , \tag{9}$$

where $z \leq \sigma$ and $\xi_i \sqsubseteq tr_i$. Using the background knowledge ξ_i, the adversary could identify a subset $TR^*(\xi_i)$ of trajectories in TR^* for which ξ_i is a sub-trajectory.

3.3 Anonymization of Trajectory Data

The greedy clustering-based approach to anonymize trajectory data consists of two main phases:

1. Clustering of trajectories based on their privacy levels.
2. Anonymization of trajectories within each cluster.

Trajectory Clustering. We can apply two different strategies for clustering of trajectories: In the first strategy, trajectories are first categorized in groups based on their privacy levels and then some clusters are created for each group independently from the others. Unfortunately, this strategy does not lead to the most effective one. This is because trajectories within one single cluster may be far removed from each other while having the same privacy level. Therefore, there will be a substantial amount of information loss in case of cluster anonymization. To illustrate this strategy better, we provide an example. Consider the set of trajectories illustrated in Fig. 1. In this figure, there are two trajectories with

privacy level 2 and four trajectories with privacy level 3. Utilizing the first strategy, all the trajectories of privacy level 3 are placed in one cluster and all the trajectories with privacy level 2 in another one. As shown, this strategy has poor performance, since the trajectory tr_4 has a long distance from its own cluster members and is much closer to another cluster.

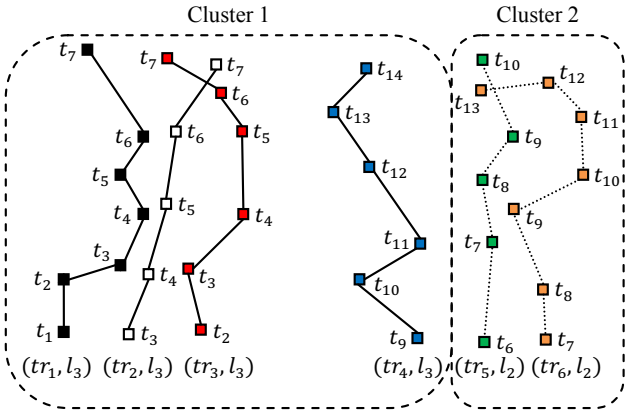


Fig. 1. Trajectory clustering using the first strategy

In the second strategy, trajectories are first categorized in groups based on their privacy levels. Clustering is then commenced from the group with maximum privacy level in a decreasing order, since preserving the privacy of trajectories with higher privacy levels has more priority. The clusters are created such that each cluster size is proportional to the privacy level of its centroid and the distance of each trajectory from the cluster centroid is less than a fixed threshold. Figure 2 illustrates this strategy. As can be seen, trajectories are partitioned into two clusters and the outlying trajectory tr_4 with privacy level 3 is placed in a cluster along with two other trajectories with privacy level 2. Therefore, this strategy significantly results in lower information loss compared with the first strategy.

Definition 7 ((δ, l)-cluster). *A (δ, l)-cluster c is a set of at least l trajectories whose distance from the cluster centroid, denoted by $\vartheta(c)$, is less than a fixed radius δ, where l is set to $l(\rho(\vartheta(c))$ and so called the privacy level of c. For simplicity, we represent the privacy level of the (δ, l)-cluster c as $l(c)$.*

Algorithm 1 shows the pseudo code of CTR that employs the second strategy to cluster trajectories. The algorithm takes the trajectory data TR, the initial cluster radius δ_{init}, and the maximum cluster radius δ_{max} as the input and returns a cluster set $\mathcal{C}$ as the output. It first initializes the cluster radius δ to δ_{init} (line 2) and the privacy level l to the maximum privacy level of trajectories in TR (lines 3–4). It then clusters trajectories in TR using an iterative process. In each iteration, PTR is applied on TR and a set $\mathcal{C}_l$ of (δ, l)-clusters is generated (line 6). Then, all trajectories in $\mathcal{C}_l$ are removed from TR and added to $\mathcal{C}$ (lines

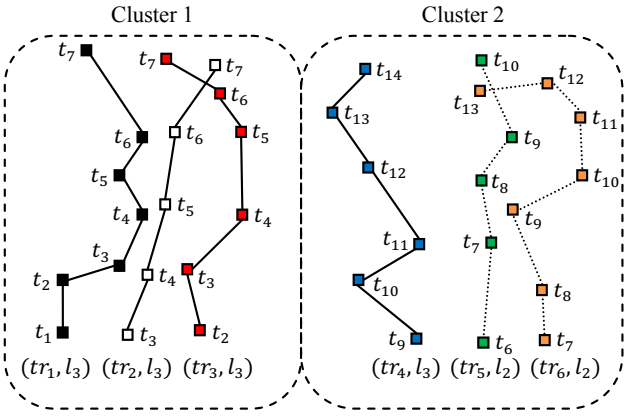


Fig. 2. Trajectory clustering using the second strategy

7–8). Finally, the value of l is decreased by 1 (line 9). This process is repeated until TR is empty or l is equal to zero (lines 5–10). There may be still some trajectories left in TR without any clusters belong to. Therefore, some clusters in $\mathcal{C}$ are enlarged with the remaining trajectories in TR. To this end, the algorithm increases the value of δ by a small amount ϵ (line 12) and for each trajectory $tr_i \in TR$ selects a set $\mathcal{D}$ of clusters in $\mathcal{C}$ whose privacy level is greater than or equal to $\ell(\rho(tr_i))$ and whose normalized trajectory distance from tr_i is less than or equal to δ (line 14). If $\mathcal{D}$ is non-empty, it selects the closest cluster c_z from the set $\mathcal{E}$ of clusters with the highest privacy level in $\mathcal{D}$ (lines 16–17). tr_i is then added to c_z and is removed from TR (lines 18–19). The above steps are repeated until TR is empty or δ is greater than δ_{max} (lines 11–22). When a trajectory cannot be added to any cluster without violating δ_{max}, it is simply ignored.

Algorithm 2 shows the pseudo code of PTR. The algorithm takes a set TR_r of trajectories, the cluster radius δ, and the privacy level l as the input and returns a (δ, l)-cluster set $\mathcal{C}_l$ as the output. It first selects a set TR_l of trajectories in TR_r with the privacy level l (line 2). It then randomly selects a trajectory $tr_c \in TR_l$ (line 4) and creates a new (δ, l)-cluster c with tr_c as its centroid (lines 5–6). It next selects a set TR_s of trajectories in TR_r whose normalized trajectory distance from $\vartheta(c)$ is less than or equal to δ (line 7). If TR_s is non-empty, it selects the closest trajectory tr_z from the trajectories with the highest privacy level in TR_s (lines 9–10). tr_z is then added to c and is removed from TR_s (lines 11–12). The above steps are repeated until TR_s is empty or $|c|$ is equal to l (lines 8–13). Clearly, $|c|$ may be still less than l that reveals $\vartheta(c)$ is an inappropriate cluster centroid. Therefore, it is removed from TR_l (lines 14–15). Nevertheless, if $|c|$ is equal to l, c is added to $\mathcal{C}_l$ and all trajectories in it are removed from TR_r and TR_l (lines 16–20). This process continues until TR_l is empty (lines 3–21).

Cluster Anonymization. Once the clusters have been created, we use ATR to anonymize trajectories within each cluster. Algorithm 3 shows the pseudo

Algorithm 1. CTR

Input:
TR: trajectory data
δ_{init}: initial cluster radius
δ_{max}: maximum cluster radius
Output:
$\mathcal{C}$: cluster set

1: $\mathcal{C} := \emptyset$
2: $\delta := \delta_{init}$
3: $\mathcal{L} := \bigcup_{tr_i \in TR} \{\ell(\rho(tr_i))\}$
4: $l := \max \mathcal{L}$
5: **while** $TR \neq \emptyset$ **and** $l > 0$ **do**
6: $\quad \mathcal{C}_l := ptr(TR, \delta, l)$
7: $\quad TR := TR - \bigcup_{c \in \mathcal{C}_l} c$
8: $\quad \mathcal{C} := \mathcal{C} \cup \mathcal{C}_l$
9: $\quad l := l - 1$
10: **end while**
11: **while** $TR \neq \emptyset$ **and** $\delta \leq \delta_{max}$ **do**
12: $\quad \delta := \delta + \epsilon$
13: $\quad$ **for** each $tr_i \in TR$ **do**
14: $\quad\quad \mathcal{D} := \{c \in \mathcal{C} \mid \ell(c) \geq \ell(\rho(tr_i)), ntd(tr_i, \vartheta(c)) \leq \delta\}$
15: $\quad\quad$ **if** $\mathcal{D} \neq \emptyset$ **then**
16: $\quad\quad\quad \mathcal{E} := \arg\max_{c \in \mathcal{D}} \ell(c)$
17: $\quad\quad\quad c_z := \arg\min_{c \in \mathcal{E}} ntd(tr_i, \vartheta(c))$
18: $\quad\quad\quad c_z := c_z \cup \{tr_i\}$
19: $\quad\quad\quad TR := TR - \{tr_i\}$
20: $\quad\quad$ **end if**
21: $\quad$ **end for**
22: **end while**
23: **return** $\mathcal{C}$

code of ATR. Let $\mathcal{C}$ be the cluster set returned by CTR. The algorithm takes $\mathcal{C}$ and the spatial threshold δ_d as the input and returns a set TR^* of anonymized trajectories as the output. For each cluster $c \in \mathcal{C}$ with centroid $\vartheta(c)$, it first selects the closest trajectory $tr_z \in c$ to $\vartheta(c)$ and removes it from c (lines 5–6). It then applies FLPT to find a sequence of joint moving points between tr_z and $\vartheta(c)$ and replaces $\vartheta(c)$ with it (line 7). The above steps are repeated for other trajectories in c until there is no trajectory left without being anonymized (lines 4–8). The algorithm next anonymizes each trajectory $tr_i \in c$ to a trajectory tr_i^* (lines 9–20). To this end, for each moving point $p_i^k \in tr_i$, if there exists a corresponding moving point $p_j^{k'} \in \vartheta(c)$ such that $p_j^{k'}$ is a linked moving point of p_i^k, *i.e.* $p_j^{k'} \in \phi(p_i^k)$, then tr_i^* is appended by $p_j^{k'}$, and otherwise, it is appended by the distorted moving point $\zeta(p_i^k, \delta_d)$ (lines 12–16). Moreover, the time coordinate of distorted moving points in tr_i^* is updated to preserve the sequentiality of moving points in tr_i^* (line 18).

Algorithm 2. PTR

Input:
TR_r: trajectory set
δ: cluster radius
l: privacy level
Output:
$\mathcal{C}_l$: (δ, l)-cluster set
1: $\mathcal{C}_l := \emptyset$
2: $TR_l := \{tr_i \in TR_r \mid \ell(\rho(tr_i)) = l\}$
3: **while** $TR_l \neq \emptyset$ **do**
4: Randomly select a trajectory $tr_c \in TR_l$
5: $c := \{tr_c\}$
6: $\vartheta(c) := tr_c$
7: $TR_s := \{tr_i \in TR_r \mid ntd(tr_i, \vartheta(c)) \leq \delta, tr_i \neq \vartheta(c)\}$
8: **while** $TR_s \neq \emptyset$ **and** $|c| < l$ **do**
9: $TR_v := \arg\max_{tr_k \in TR_s} \ell(\rho(tr_k))$
10: $tr_z := \arg\min_{tr_k \in TR_v} ntd(tr_k, \vartheta(c))$
11: $c := c \cup \{tr_z\}$
12: $TR_s := TR_s - \{tr_z\}$
13: **end while**
14: **if** $|c| < l$ **then**
15: $TR_l := TR_l - \{\vartheta(c)\}$
16: **else**
17: $\mathcal{C}_l := \mathcal{C}_l \cup \{c\}$
18: $TR_r := TR_r - c$
19: $TR_l := TR_l - c$
20: **end if**
21: **end while**
22: **return** $\mathcal{C}_l$

Algorithm 4 shows the pseudo code of FLPT. The input to the algorithm is a pair of trajectories tr_i and tr_j. It first defines a matrix s of elements. Each element $s[k, k']$ holds a sequence of joint moving points between two sub-trajectories $tr_i^k = \langle p_i^1, \ldots, p_i^k \rangle$ and $tr_j^{k'} = \langle p_j^1, \ldots, p_j^{k'} \rangle$. It then defines the neighbor set $g^{k,k'}$ as the set of elements that are vertically and horizontally adjacent to $s[k, k']$. In each iteration of nested loops (lines 8–19), if p_i^k is a linked moving point of $p_j^{k'}$, *i.e.* $p_i^k \in \phi(p_j^{k'})$, the element $s[k, k']$ is substituted by its diagonal neighbor appended by the joint moving point $p_{ij}^{k,k'}$ computed as the average of $p_i^k \in tr_i^k$ and $p_j^{k'} \in tr_j^{k'}$ (lines 10–12). Otherwise, the element $s[k, k']$ is substituted by the neighbor with maximum number of joint moving points (lines 13–17). At the end of the algorithm, the whole sequence of joint moving points between two trajectories tr_i and tr_j, $s[|tr_i|, |tr_j|]$, is assigned to tr_v (line 20).

Example 1 (Joint moving points). Consider the following two trajectories:

$$tr_i = \langle (2, 120, 70), (4, 125, 79), (6, 131, 82), (8, 133, 90), (10, 135, 95), (12, 137, 98) \rangle$$

Algorithm 3. ATR

Input:
$\mathcal{C}$: cluster set
δ_d: spatial threshold
Output:
TR^*: trajectory data

$TR^* := \emptyset$
for each cluster $c \in \mathcal{C}$ **do**
  $c := c - \{\vartheta(c)\}$
  while $c \neq \emptyset$ **do**
    $tr_z := \arg\min_{tr_k \in c} ntd(tr_k, \vartheta(c))$
    $c := c - \{tr_z\}$
    $\vartheta(c) := flpt(tr_z, \vartheta(c))$
  end while
  for each trajectory $tr_i \in c$ **do**
    $tr_i^* := \langle\rangle$
    for each moving point $p_i^k \in tr_i$ **do**
      if there exists a moving point $p_j^{k'} \in \vartheta(c)$ such that $p_j^{k'} \in \phi(p_i^k)$ **then**
        $tr_i^* := append(tr_i^*, \langle p_j^{k'} \rangle)$
      else
        $tr_i^* := append(tr_i^*, \zeta(p_i^k, \delta_d))$
      end if
    end for
    Update time coordinate of distorted moving points in tr_i^*
    $TR^* := TR^* \cup \{tr_i^*\}$
  end for
end for
return TR^*

$$tr_j = \langle (3, 126, 74), (5, 138, 80), (7, 140, 88), (9, 130, 79) \rangle$$

Given the temporal and spatial thresholds $(\delta_t, \delta_d) = (1, 8)$, the matrix s for tr_i and tr_j is computed as shown in Fig. 3. As can be seen, $p_i^2 \in tr_i$ is a linked moving point of $p_j^1 \in tr_j$. Therefore, the element $s[2,1]$ is set to $append(s[1,0], \langle p_{ij}^{2,1} \rangle)$. On the contrary, $p_i^4 \in tr_i$ is not a linked moving point of $p_j^2 \in tr_j$. Thus, the element $s[4,2]$ is substituted by the element $s[3,2]$ that is the neighbor with maximum number of joint moving points. At the end, the element $s[6,4]$ holds the whole sequence of joint moving points between tr_i and tr_j, *i.e.*, $\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3} \rangle$.

4 Experiments

In this section, we present the experimental results evaluating the performance of our approach. The experiments were run on a PC with a 2.00 GHz Core 2 Duo Intel processor and 2 GB RAM. We implemented our approach in C# with Microsoft Visual Studio 2010.

Algorithm 4. FLPT

Input:
tr_i, tr_j: trajectory
Output:
tr_v: trajectory

```
1: s[0,0] := ⟨⟩
2: for k := 1 to |tr_i| do
3:     s[k,0] := ⟨⟩
4: end for
5: for k' := 1 to |tr_j| do
6:     s[0,k'] := ⟨⟩
7: end for
8: for k := 1 to |tr_i| do
9:     for k' := 1 to |tr_j| do
10:        if p_i^k ∈ φ(p_j^k') then
11:            p_ij^{k,k'} := (p_i^k + p_j^k')/2
12:            s[k,k'] := append(s[k-1,k'-1], ⟨p_ij^{k,k'}⟩)
13:        else
14:            g^{k,k'} := {s[k-1,k'], s[k,k'-1]}
15:            g_max^{k,k'} := arg max_{x∈g^{k,k'}} |x|
16:            s[k,k'] := g_max^{k,k'}
17:        end if
18:    end for
19: end for
20: tr_v := s[|tr_i|, |tr_j|]
21: return tr_v
```

		p_j^1	p_j^2	p_j^3	p_j^4
	$\langle\rangle$	$\langle\rangle$	$\langle\rangle$	$\langle\rangle$	$\langle\rangle$
p_i^1	$\langle\rangle$	$\langle p_{ij}^{1,1}\rangle$	$\langle p_{ij}^{1,1}\rangle$	$\langle p_{ij}^{1,1}\rangle$	$\langle p_{ij}^{1,1}\rangle$
p_i^2	$\langle\rangle$	$\langle p_{ij}^{2,1}\rangle$	$\langle p_{ij}^{1,1}\rangle$	$\langle p_{ij}^{1,1}\rangle$	$\langle p_{ij}^{1,1}\rangle$
p_i^3	$\langle\rangle$	$\langle p_{ij}^{2,1}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}\rangle$
p_i^4	$\langle\rangle$	$\langle p_{ij}^{2,1}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3}\rangle$
p_i^5	$\langle\rangle$	$\langle p_{ij}^{2,1}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3}\rangle$
p_i^6	$\langle\rangle$	$\langle p_{ij}^{2,1}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3}\rangle$	$\langle p_{ij}^{2,1}, p_{ij}^{3,2}, p_{ij}^{4,3}\rangle$

Fig. 3. The matrix s computed using FLPT for two trajectories tr_i and tr_j

4.1 Dataset

We used 1777 synthetic trajectories generated with Brinkhoff's network-based generator [17] representing moving objects' movements over the road-network of the German city of Oldenburg. The total number of moving points was 100000 generated during 145 timestamps and the maximum trajectory length was 145. The difference between two consecutive timestamps was 10 minutes.

4.2 Experimental Results

In this section, we evaluate the performance of our approach in terms of total space distortion, re-identification probability, and average run time.

Space Distortion. The main objective of our approach is to keep anonymized trajectory data as close to the original one as possible. We use total space distortion to capture the distortion of all anonymized trajectories from original ones.

Definition 8 (Total space distortion). *The space distortion of an anonymized trajectory tr_i^* with respect to its original trajectory tr_i is defined as*

$$d(tr_i, tr_i^*) = \begin{cases} \Omega.|tr_i| & |tr_i^*| = 0 \ , \\ \min\{d(\theta(tr_i), \theta(tr_i^*)) + \varrho(p_i^1, p_i^{1*}), & \\ \quad d(tr_i, \theta(tr_i^*)) + \Omega, & \text{otherwise} \ , \\ \quad d(\theta(tr_i), tr_i^*) + \Omega\} & \end{cases} \tag{10}$$

where $\varrho(p_i^1, p_i^{1*}) = \Delta(p_i^1, p_i^{1*})$ if $p_i^{1*} \in \phi(p_i^1)$ and $\varrho(p_i^1, p_i^{1*}) = \Omega$ otherwise. Δ is the Euclidean distance and Ω is a constant that penalizes distorted moving points. Ω is set to a value greater than δ_d which is the maximum Euclidean distance between two linked moving points. Let TR be the original trajectory data. The total space distortion of TR from its anonymized trajectory data TR^* is defined as

$$\mathcal{D}_{total}(TR, TR^*) = \sum_{tr_i \in TR} d(tr_i, tr_i^*) \ , \tag{11}$$

where $tr_i^* \in TR^*$ is the anonymized version of $tr_i \in TR$.

Figure 4 shows the total space distortion, $\mathcal{D}_{total}(TR, TR^*)$, after anonymizing the Oldenburg dataset for different privacy levels, $l = 1, \ldots, 5$ and $l = 1, \ldots, 7$. As expected, trajectories with higher privacy levels have higher amounts of space distortion. This is because, the more the privacy level, the larger the cluster size will be. Thus, we need a large amount of distortion to anonymize trajectories within each cluster.

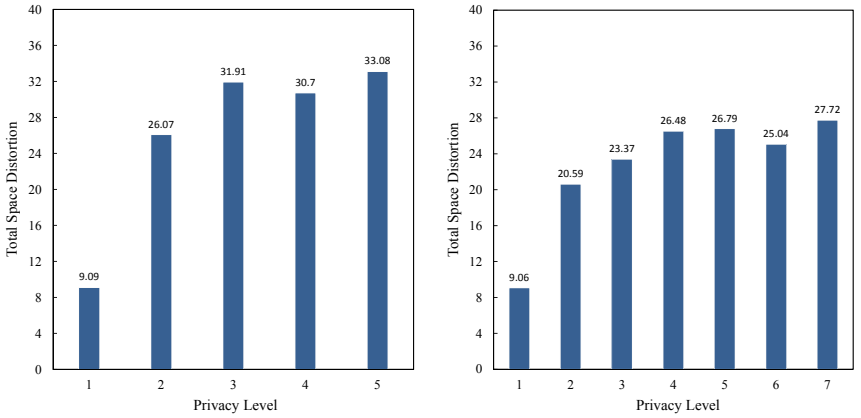


Fig. 4. Total space distortion in millions for $\delta = 0.4$ and different privacy levels, (a) $l = 1, \ldots, 5$, and (b) $l = 1, \ldots, 7$

Re-identification Probability. We use re-identification probability to measure the probability of privacy breach of anonymized trajectory data. It represents the maximum probability of re-identification of a trajectory by an adversary in anonymized trajectory data having some background knowledge.

Definition 9 (Re-identification probability). *Let $tr_i^* \in TR^*$ be the anonymized version of a trajectory $tr_i \in TR$. The re-identification probability of tr_i given some adversary's background knowledge $\xi_i \sqsubseteq tr_i$ about $\rho(tr_i)$ is calculated as*

$$P_r^M(tr_i \mid \xi_i) = \max_{\xi_k \sqsubseteq \xi_i} P_r(tr_i \mid \xi_k) \ , \tag{12}$$

$$P_r(tr_i \mid \xi_k) = \begin{cases} 1/(|TR^*(\xi_k)|) & \xi_k \sqsubseteq tr_i^* \ , \\ 0 & \text{otherwise} \ , \end{cases} \tag{13}$$

where $TR^*(\xi_k)$ is a subset of trajectories in TR^* for which ξ_k is a sub-trajectory.

The re-identification probability depends on the length of the adversary's background knowledge, *i.e.*, the number of moving points it contains. Since some moving points of original trajectories might be distorted due to the cluster anonymization phase, we need to consider ξ_i and all its possible sub-trajectories $\xi_k \sqsubseteq \xi_i$ in calculating $P_r^M(tr_i \mid \xi_i)$.

Figure 5 shows the re-identification probability, $P_r^M(tr_i \mid \xi_i)$, for different lengths of the adversary's background knowledge, $|\xi_i| = 1, \ldots, 10$, and different privacy levels, $l = 1, \ldots, 5$ and $l = 1, \ldots, 7$. The experiments were run 100 times for each parameter setting and the average results are reported. As can be seen, the re-identification probability grows slightly with increasing the length of the adversary's background knowledge.

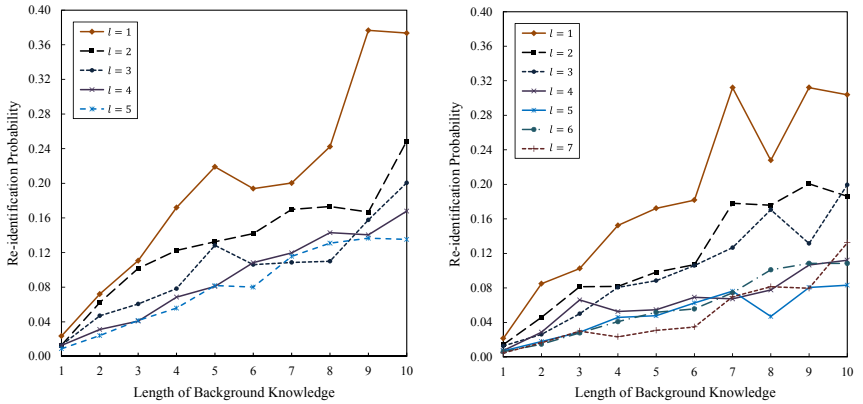


Fig. 5. Re-identification probability for different lengths of the adversary's background knowledge, $|\xi_i| = 1, \ldots, 10$, and different privacy levels, (a) $l = 1, \ldots, 5$ and (b) $l = 1, \ldots, 7$

Figure 6 shows the average re-identification probability of different lengths of the adversary's background knowledge for different privacy levels, $l = 1, \ldots, 5$ and $l = 1, \ldots, 7$. As expected, trajectories with higher privacy levels have lower amounts of re-identification probability, which results in lower probability of privacy breach.

It is important to note that in the above experiments the value of δ was set to 0.4.

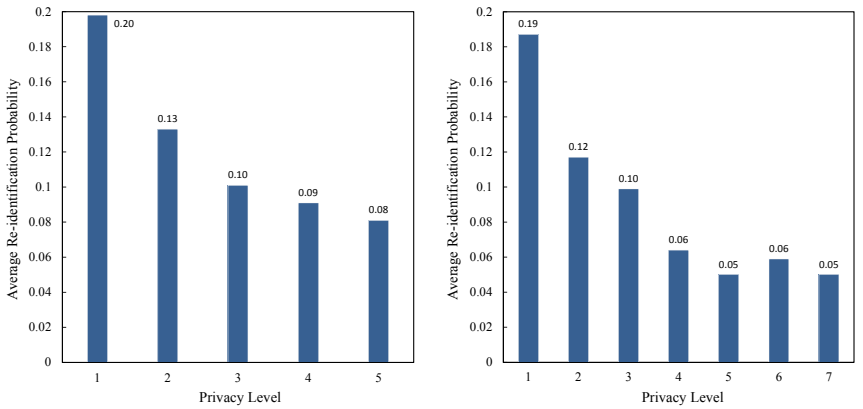


Fig. 6. Average re-identification probability for different privacy levels, (a) $l = 1, \ldots, 5$ and (b) $l = 1, \ldots, 7$

Run Time. Time performance of every anonymization algorithm is at a premium due to inherent complexity of trajectory data. Figure 7 shows the average run time of our approach on the Oldenburg dataset for different values of δ and l_{max}. As it is clear from the figure, the average run time drops substantially by increasing the value of δ. However, after $\delta = 0.6$ the average run time keeps on decreasing more steadily. It should be mentioned that as the value of δ increases, the time of clustering phase and consequently the average run time of our approach decreases.

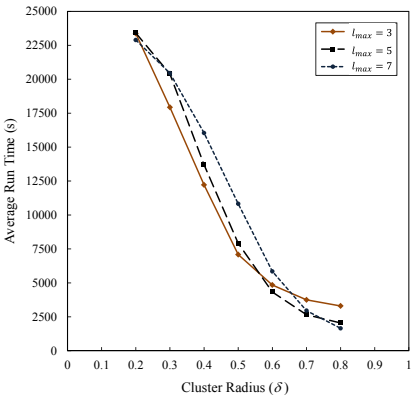


Fig. 7. Average run time in seconds for different values of δ and l_{max}

5 Conclusion

Many existing approaches for trajectory anonymization consider the same levels of privacy for different moving objects. In this paper, we proposed a greedy clustering-based approach for anonymizing trajectory data in which the privacy requirements of moving objects are not considered to be the same. We discussed different phases of our approach including trajectory clustering and cluster anonymization. We described two different strategies for clustering of trajectories based on their privacy levels. We further presented a novel matching point algorithm for cluster anonymization that takes the sequentiality of moving points into account while finding linked moving points between two trajectories. We used a dataset of synthetic moving object trajectories and evaluated the performance of our approach in terms of total space distortion, re-identification probability, and average run time.

Acknowledgments. This work was supported in part by the Iran Telecommunication Research Center (ITRC) under contract number 500/15716.

References

1. Frentzos, E., Gratsias, K., Pelekis, N., Theodoridis, Y.: Nearest Neighbor Search on Moving Object Trajectories. In: Medeiros, C.B., Egenhofer, M., Bertino, E. (eds.) SSTD 2005. LNCS, vol. 3633, pp. 328–345. Springer, Heidelberg (2005)
2. Lee, J.-G., Han, J., Li, X., Gonzalez, H.: raClass: Trajectory Classification Using Hierarchical Region-based and Trajectory-based Clustering. In: Proc. of the 34th Int. Conf. on Very Large Databases (VLDB 2008), Auckland, New Zealand (2008)
3. Lee, J.-G., Han, J., Whang, K.-Y.: Trajectory Clustering: a Partition-and-Group Framework. In: Proc. of the ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 2007), Beijing, China, pp. 593–604 (2007)
4. Li, X., Han, J., Kim, S., Gonzalez, H.: Anomaly Detection in Moving Object. In: Chen, H., Yang, C.C. (eds.) Intelligence and Security Informatics. SCI, vol. 135, pp. 357–381. Springer, Heidelberg (2008)
5. Pelekis, N., Kopanakis, I., Kotsifakos, E.E., Frentzos, E., Theodoridis, Y.: Clustering Trajectories of Moving Objects in an Uncertain World. In: Proc. of the 9th IEEE Int. Conf. on Data Mining (ICDM 2009), Miami, USA, pp. 417–427 (2009)
6. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data 1(1) (2007)
7. Samarati, P.: Protecting Respondents Privacy in Microdata Release. IEEE Transactions on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
8. Sweeney, L.: k-Anonymity: a Model for Protecting Privacy. International Journal of Uncertainty, Fuzziness Knowledge-Based Systems 10(5), 557–570 (2002)
9. Chen, L., Özsu, M.T., Oria, V.: Robust and Fast Similarity Search for Moving Object Trajectories. In: Proc. of the 24th ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 2005), Maryland, USA, pp. 491–502 (2005)
10. Terrovitis, M., Mamoulis, N.: Privacy Preservation in the Publication of Trajectories. In: Proc. of the 9th Int. Conf. on Mobile Data Management (MDM 2008), Beijing, China, pp. 65–72 (2008)
11. Yarovoy, R., Bonchi, F., Lakshmanan, L.V.S., Wang, W.H.: Anonymizing Moving Objects: How to Hide a MOB in a Crowd? In: Proc. of the 12th Int. Conf. on Extending Database Technology (EDBT 2009), Saint Petersburg, Russia, pp. 72–83 (2009)
12. Nergiz, E., Atzori, M., Saygin, Y.: Towards Trajectory Anonymization: a Generalization-Based Approach. Transactions on Data Privacy 2(1), 47–75 (2009)
13. Abul, O., Bonchi, F., Nanni, M.: Anonymization of Moving Objects Databases by Clustering and Perturbation. Information Systems 35(8), 884–910 (2010)
14. Abul, O., Bonchi, F., Nanni, M.: Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases. In: Proc. of the 24th IEEE Int. Conf. on Data Engineering (ICDE 2008), Cancun, Mexico, pp. 376–385 (2008)
15. Monreale, A., Andrienko, G., Andrienko, N., Giannotti, F., Pedreschi, D., Rinzivillo, S., Wrobel, S.: Movement Data Anonymity through Generalization. Transactions on Data Privacy 3(2), 91–121 (2010)
16. Bonchi, F.: Privacy Preserving Publication of Moving Object Data. In: Bettini, C., Jajodia, S., Samarati, P., Wang, X.S. (eds.) Privacy in Location-Based Applications. LNCS, vol. 5599, pp. 190–215. Springer, Heidelberg (2009)
17. Brinkhoff, T.: Generating Traffic Data. IEEE Data Engineering Bulletin 26(2), 19–25 (2003)

Estimating the Number of Hosts Corresponding to an Address while Preserving Anonymity

Alif Wahid, Christopher Leckie, and Chenfeng Zhou

Department of Computing and Information Systems
The University of Melbourne, Australia
a.wahid@student.unimelb.edu.au, caleckie@unimelb.edu.au, cvzhou@gmail.com

Abstract. Estimating the number of hosts that have been assigned to an Internet address is a challenging problem due to confounding factors such as the dynamic allocation of addresses and the prohibition of access to privacy sensitive data that can reveal user identities and remove anonymity. We propose a probabilistic method that strikes a desired balance between *protection of anonymity* and *accuracy of estimation*. By utilising the phenomenon of preferential attachment, we show that the number of hosts corresponding to an address is accurately predicted by the number of times that an address appears in a series of alternating ON and OFF intervals. We validate our method using a four month trace of dynamic address allocations at a campus wireless network. In so doing, we demonstrate the practical significance and utility of such an anonymity preserving method for estimating the number of hosts corresponding to a dynamic address.

1 Introduction

IP addresses are a limited resource for hosts connected to the Internet. This stems from the successful deployment of the IPv4 protocol [10] over the course of three decades, which also had the unforeseen consequence of exhausting a limited pool of globally unique addresses [18]. A large number of control plane protocols have also been deployed over the years to support the rapid growth of IPv4 infrastructure throughout the world. These include dynamic address assignments [6, 22], translation of addresses at network gateways [24], domain name look-ups for decoupling addresses from servers [17], application layer proxies, and firewalls for security and privacy protection. As a result, IP addresses are seldom tightly bound to individual hosts for long periods in the order of weeks or months, since they are necessarily recycled and/or shared by multiple hosts. This aliasing phenomenon raises a fundamental challenge from the perspective of network monitoring: *how can we reliably infer the number of hosts corresponding to an address?* The difficulty of this challenge is further highlighted by the legally binding requirement of protecting the anonymity of all users associated with any given host such that observations of an address must never be *unambiguously* linked with human activity that could reveal identities and remove privacy (refer to the "Information Privacy Principles" legislated in the authors' jurisdiction [25]).

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 166–179, 2012.

Underlying this challenge is the subsequent question of *why is the number of hosts corresponding to an address an important quantity?* Internet hosts generate and absorb all traffic that one can measure at the network layer. This means that any uncertainty underlying the population of hosts due to dynamic addressing has a pervasive impact on all analysis and inference tasks involved in network measurement and traffic engineering. Moreover, in the context of network security and monitoring, one needs reliable ways of identifying, tracking and eliminating those Internet hosts that engage in deliberate malicious activity and cyber-crime over the course of many weeks and months, e.g., spamming mail servers, Distributed Denial of Service (DDoS) botnets, port-scanning worms, and so on. In these cases, accurately estimating the number of hosts corresponding to an address is of real practical significance since the threat scales with the population of malicious hosts and not the number of unique addresses [21]. But the requirement of anonymity protection still remains, which severely restricts access to any data that might improve the accuracy of estimation (e.g., Ethernet MAC addresses that are tightly bound to their corresponding hosts). Therefore, any method of estimating the number of hosts corresponding to an address must find a trade-off that preserves user anonymity as well as providing accurate and reliable estimates of the host population.

In this paper, we propose a solution to this problem and demonstrate its validity using real-world data. Our main contributions are as follows. First, we develop a stochastic model of host to address binding that is dependent on one hidden variable and one observed variable. This model is discriminative (as opposed to generative) in the sense that it only allows the calculation of conditional probabilities for the number of hosts corresponding to an address given: a) the number of times the address appeared, and b) its latent probability of binding to a new host in each appearance. Second, we derive and test careful approximations of this model using a large data set of dynamic host to address bindings from a campus wireless network that comes with anonymised ground truth regarding the exact number of hosts corresponding to an observed address. However, this ground truth is purely for validation and our model is only dependent on the two stated parameters related to the observed addresses (as opposed to parameters related to the hidden hosts, which potentially violate user anonymity). Thus, we have developed a probabilistic framework that can readily cope with the underlying uncertainty that arises from anonymity protection, while still providing an accurate and reliable estimate for the population of hosts corresponding to a dynamic address.

The rest of this paper is structured as follows. We begin by detailing some of the motivating use cases and operational context that make this a practically important problem in Sect. 2. Then Sect. 3 derives our proposed model and presents various implications. We develop approximations of this model in order to efficiently estimate the corresponding number of hosts for large numbers of addresses in Sect. 4. These methods are subsequently tested and demonstrated in Sect. 5 using a four month trace of sanitised DHCP logs collected from a

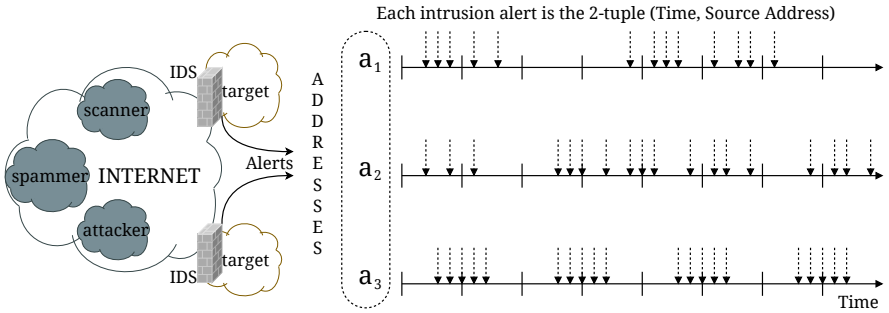


Fig. 1. An example scenario of intrusion alerts that need to be unambiguously mapped to malicious hosts for reliable blacklisting and/or other persistent countermeasures

campus wireless network. Finally, we discuss the relevant literature in Sect. 6 before concluding the paper in Sect. 7.

2 Operational Context

Consider the network security scenario of Fig. 1. A private network operator who happens to be among the targets of malware propagating on the Internet can install an Intrusion Detection System (IDS) to automatically monitor the transport layer packet-flows entering and leaving their network. Any packet-flow found to be malicious according to prespecified criteria can be flagged and its corresponding source address blacklisted as a typical countermeasure. However, due to the temporal volatility of the binding between a malicious host and its corresponding address, such intrusion alerts are necessarily unreliable. Refer to [9] for an example of Fast-Flux Service Networks that deliberately exploit such volatility to remain undetected for considerable periods. In addition, a common form of attack scenario involves spoofing addresses due to the unaccountability of the IPv4 protocol suite [1], which only accentuates the aliasing and uncertainty of host to address binding. Hence, the process of blacklisting will eventually lead to adverse consequences whereby legitimate users are denied access (who could potentially be paying customers of a cloud service for instance) [27]. A method of unambiguously inferring the binding between an alerted address and its corresponding malicious host(s) is required to reliably determine an expiration date for the blacklist.

Another motivating use case is of resource provisioning in mobile and wireless networks that are required to recycle a limited pool of addresses in order to accommodate an oversubscribed set of hosts for access to essential services like web proxies and Network Attached Storage (NAS). Typically, a DHCP server is the first point of contact for mobile hosts that are attempting to access those services. Even though the DHCP protocol relies on the static binding between a host Operating System (OS) and its corresponding Ethernet MAC address [6], that information is generally prohibited from being used for any other subsequent

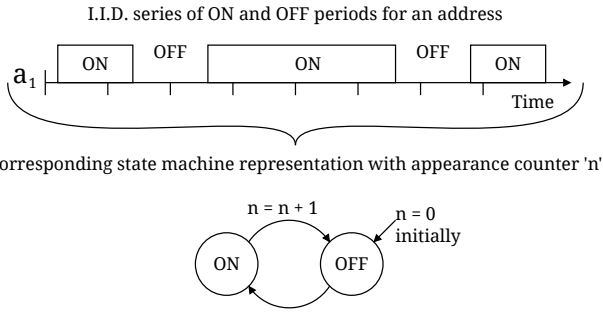


Fig. 2. Traditional model of a traffic source (i.e., an address) as an alternating series of independent ON and OFF periods. The time-series representation (top) can subsequently be translated into a state machine representation (bottom) that counts the number of discrete "appearances" by an address.

purpose, e.g., retrospective inference of host to address bindings such that the lease time of individual addresses can be fine-tuned in order to maximise the utilisation of the small pool of private addresses [11]. Such network operational and management proposals raise legal barriers due to the privacy protection requirements in various jurisdictions [25]. The challenge in these use cases is to reliably infer the population of hosts corresponding to a dynamic address without knowing the exact temporal binding between hosts and addresses. In the next two sections of this paper, we tackle precisely this problem.

3 Our Model of Host to Address Binding

A network traffic source is traditionally modelled as a series of alternating "ON" and "OFF" periods, which are independent and identically distributed (i.i.d.) [14, 19]. This is illustrated in Fig. 2. Statistical self-similarity and long-range dependence can emerge if the ON or the OFF periods exhibit a heavy-tailed distribution with infinite variance [23,28]. From the perspective of network measurement, this manifests as bursty traffic from any given address at any given time-scale that can be clustered into discrete sequences of contiguous ON and OFF periods. These ON and OFF periods are distributed according to a power-law probability density that is parameterised by a scaling exponent and a tail threshold [5]. The question that we are interested in is *how can this well-established fractal phenomenon be exploited to derive an anonymity preserving model of host to address binding?*

The insight that we draw upon is the concept of *preferential attachment* [2], whereby an address is solely attached to one host during any given ON period, and subsequently, the state transition (as per Fig. 2) from one ON period to the next provides an opportunity for it to attach to either a brand new host or another host that it attached to in the past. As a result, the discrete *appearances* of an address become a measurable quantity that allows us to hypothesise

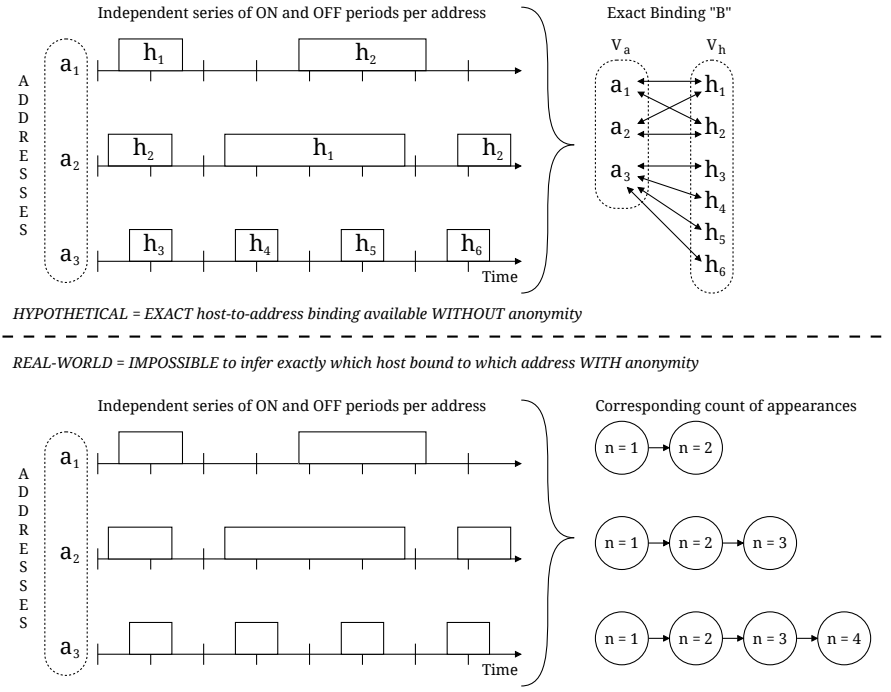


Fig. 3. Illustration of how addresses can attach or bind to hosts, and vice versa

about the number of hosts corresponding to that address over the course of an observation window. Formally, we define a host-to-address binding as a bipartite graph $B = (V_h, V_a, E_t)$, where V_h is the set of vertices corresponding to hosts, V_a is the set of vertices corresponding to addresses, and E_t is the set of edges that connects hosts to addresses in an aggregate manner over time. We generally denote the number of hosts by h and an individual address by a with numerical subscripts to differentiate further. On the top right-hand corner of Fig. 3, we depict the host-to-address binding corresponding to the example time-series on the top left-hand side. In the absence of any data that would easily give away the exact binding B, we can only attempt an estimation on probabilistic grounds by using the measurable number of appearances, denoted n, for any given address a. This uncertainty is illustrated in the bottom half of Fig. 3 where the labelled host attachments have been removed such that one cannot distinguish exactly which host bound to which address for how long and exactly when.

The notion of preferential attachment can now be formally expressed by the probability $P(h_a \leq k|n, p_n)$, where h_a is the number of hosts corresponding to an address $a \in V_a$, $k \in \{1, 2, 3, ...n\}$ and p_n is a hidden parameter that is to be estimated from observations. It specifies the finite independent probability with which an address $a \in V_a$ may bind to a new host at its n^{th} appearance (conversely, $\bar{p}_n = 1 - p_n$ specifies the probability that the address may bind to a previously attached host at the n^{th} appearance). So what this cumulative

distribution function (CDF) encompasses is the deviation of the binding B from the non-aliasing expectation of static address binding, whereby $h_a = 1\ \forall a \in V_a$ and $|V_a| = |V_h|$. Moreover, the monotonically increasing nature of an address's appearance count n implies that those addresses that have already attached to multiple hosts in the past are *preferred* by new hosts that may appear in the future. We formally define the estimation problem addressed in this paper as follows.

Definition 1. *Given a series of alternating ON and OFF intervals corresponding to each address $a \in V_a$, then the conditionally expected number of hosts per address, $E[h|n, p_n]$, for the underlying binding B is:*

$$E[h|n, p_n] = \frac{1}{|V_a|} \sum_{a \in V_a} \sum_{k \in n} k \cdot P(h_a = k|n, p_n) \tag{1}$$

The main assumption of our model so far is that of independence: both between addresses and between appearances. This means that any given address traces out a random path from a full binary tree that is n levels deep. Fig. 4 demonstrates one such address with $n = 4$ appearances, which has a corresponding parameter vector $p_n = \{p_1, p_2, p_3, p_4\}$ specifying its binding probabilities for *new hosts* at the n^{th} appearance, and conversely, the parameter vector $\bar{p}_n = \{\bar{p}_1, \bar{p}_2, \bar{p}_3, \bar{p}_4\}$ specifying the binding probabilities for *old hosts* at the n^{th} appearance. As is evident, the first appearance can only mean one host and so we can expect p_1 to be 1 with its complement being 0. However, the subsequent appearances must diverge in a binary manner depending on whether a new host or an old host is attached to the address. Thus, after the fourth appearance we have eight separate possibilities that arise due to the assumption of independence between appearances (note that in some cases the same number of hosts is inferred via mutually independent paths). Therefore, the space of possibilities for the number of hosts corresponding to an address grows exponentially with the number of times that the address appears during the observation window. In fact, over the course of many weeks and months it would be perfectly normal for an address to appear hundreds of times, which means that the running time complexity of computing equation (1) is in $O(|V_a| \cdot 2^n)$. This necessitates finding an approximation in order to make the computation tractable for large data sets with thousands of independent addresses and their corresponding appearance counts spanning more than two orders of magnitude.

4 Approximating the Binding Probabilities

Our approach to approximating the exponentially growing space of possibilities for one address is to assume that the binding probabilities are constants for any given number of appearances. That is to say that we reduce the full binary tree comprising a vector of n independent binding probabilities to the Binomial distribution with one binding probability that covers all the paths of the tree for a corresponding n. In other words, we have a forest of n fixed depth binary

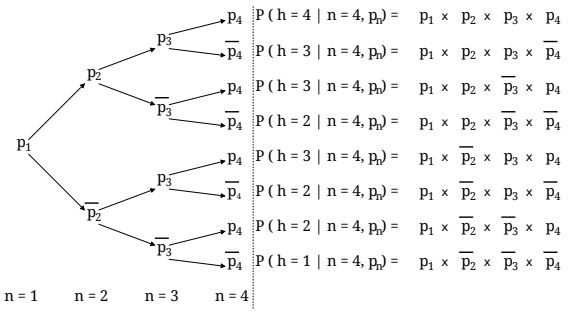


Fig. 4. The complete space of stochastic paths for $n = 4$ appearances by an address and the resulting conditional probabilities for the number of hosts in each case

trees that are individually indexed to Binomial distributions for computing the conditional probability of the number of hosts corresponding to any given address. As a result, the illustrated scenario of Fig. 4 and all other cases can be approximated by

$$P(h = k|n, p_n) = \binom{n}{k}(p_n)^k(1 - p_n)^{n-k} \tag{2}$$

for any fixed $p_n \in (0, 1]$ and $k \in \{1, 2, 3, ..., n\}$. However, the main issue that remains unsolved is how to estimate this fixed p_n for any given n with minimal loss of information. That is to say that we require a mapping function $f : \mathbb{N} \rightarrow (0, 1]$ that exploits any inherent relationship (e.g., correlation) between n and p_n that may exist.

We propose two variants of this mapping function for empirically testing whether there is indeed a relationship between n and p_n. The first is a power-law function of the form

$$p_n = c_1 \cdot n^{-\alpha} \tag{3}$$

where α is a free parameter in $(0, 1]$ and c_1 is a scaling constant in $(0, 1]$. This means that we can conjecture a correlation between p_n and n of the following form

$$\ln(p_n) = \ln(c_1) - \alpha \ln(n) \tag{4}$$

that can be tested via linear regression in the log-log domain assuming log-normally distributed error (which is not at all unreasonable in this case [16]). The second variant is an exponential function of the form

$$p_n = c_2 \cdot e^{-\lambda n} \tag{5}$$

where λ is a strictly positive free parameter and c_2 is a strictly positive scaling constant. This allows us to conjecture a correlation between p_n and n of the following form,

$$\ln(p_n) = \ln(c_2) - \lambda n \tag{6}$$

which can also be tested via linear regression in the log-linear domain assuming log-normally distributed error.

These two variants have different implications although both contain one free parameter, which is essential since any more would be a classic case of over-fitting an already explicit approximation. The power-law variant allows a slower decay than the exponential form. This means that as an address appears more often, we inherently expect it to have an increasing number of hosts due to preferential attachment, but the rate of this increase is what differs between the two variants. This is apparent when we substitute the two ways of mapping p_n while calculating the mean of the Binomial distribution as follows.

$$\mu_\alpha = n \cdot p_n = n \cdot c_1 \cdot n^{-\alpha} \tag{7}$$

$$\mu_\lambda = n \cdot p_n = n \cdot c_2 \cdot e^{-\lambda n} \tag{8}$$

Equations (7) and (8) also provide a consistency check by ensuring that the mapping functions can be readily substituted into the Binomial distribution in place of p_n without leading to any singularities. The equation for the variance of the Binomial distribution also satisfies substitution of these functions.

5 Evaluation Using DHCP Logs

In order to test our proposed approximations and the validity of the underlying model, we used a set of sanitised DHCP logs [26] collected at a campus wireless network over the course of nearly four months from midnight 19th of April 2005 to midnight 8th of August 2005. Each row in this data set contains three fields: 1) UTC time-stamp, 2) 32-bit private IP address, and 3) 48-bit anonymised MAC address. These fields collectively specify the instant when an independent IP address was "leased" by an anonymous host (as identified by its MAC address). Each unique time-stamp corresponding to a recurring IP address directly specifies a new appearance. Moreover, each new appearance can attach either to a new host or an old host as determined via the corresponding set of MAC addresses. So the total number of appearances by an IP address is simply the cardinality of its corresponding set of time-stamps. We found a total of 6,554 independent IP addresses in these logs that were leased multiple times by over 7,000 independent hosts. This resulted in an over-subscription rate of around 10% during the course of the measurement interval, which is consistent with other studies of DHCP logs [11]. We filtered the raw data down to 6,033 addresses by ensuring that for any given number of appearances there were at least three corresponding independent addresses in order to form a sufficiently large sample (Fig. 5) and increase the statistical power of our hypothesis testing.

The results of linear regressions involving the power-law and the exponential mapping functions between empirical $\hat{p}_n$ and the observed n are plotted in Figures 6, 7, 8 and 9. Note that the empirical $\hat{p}_n$ values were estimated from the data by taking the ratio of the true number of hosts corresponding to an IP

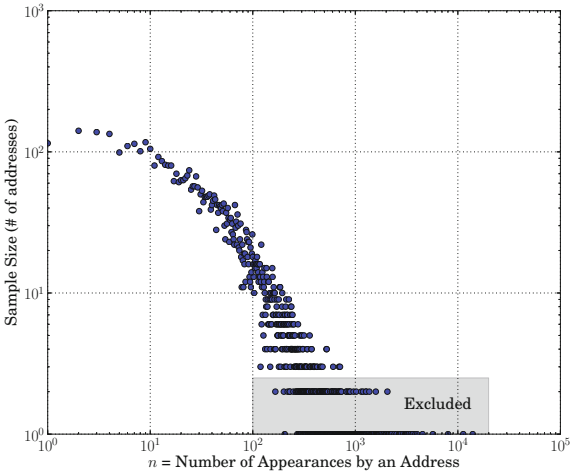


Fig. 5. Distribution of sample sizes, as measured by the number of independent addresses for each value of n. Note that in our regression analysis we excluded all the samples where there were less than three corresponding independent addresses.

address and its appearance count n, which is generally an unbiased estimator of the Binomial distribution's mean. For each sample comprising a set of independent addresses, all having the same appearance count n, we used both the mean and the median $\hat{p}_n$ to be robust and objective in our hypothesis testing.

In all four figures, the best-fit linear regressions were statistically significant at the 0.01 level assuming normally distributed errors in the logarithm domain. This means that we cannot reject either of the two mapping variants even though, upon closer scrutiny, the power-law function is a better fit than the exponential function according to the coefficient of determination r^2. In the best case, our approximation can explain more than 70% of the variance between the data and the proposed model (Fig. 6) while in the worst case this drops to around 50%. The power-law mapping appears to fit very accurately in the region $1 \leq n \leq 10$ where we have consistently large samples in the vicinity of one hundred addresses. They both appear to fit the tail of the data equally well for $n \geq 100$, which is where the error is most apparent due to small samples of less than ten independent addresses (as evident from Fig. 5).

The existence of two clear, overt correlations between p_n and n is the major finding of our evaluation, which validates the approach we took in approximating our proposed model. As a result, we can use either of these correlations to compute a conditional probability distribution for the number of hosts $P(h = k|n, p_n)$ by substituting back into equation (2). The validity of the relationship between n and p_n means that the expected value of this distribution can be a reliable estimate for the typical number of hosts per address in real-world data sets that either

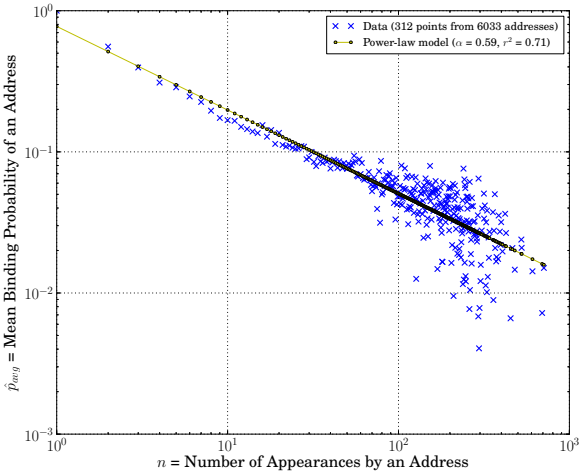


Fig. 6. Power-law mapping between n and the mean p_n of the corresponding sample

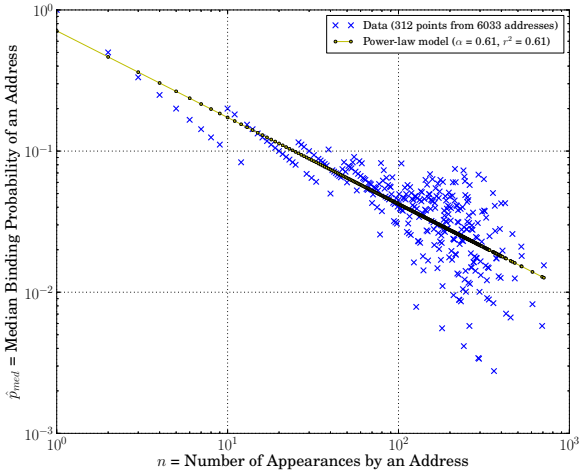


Fig. 7. Power-law mapping between n and the median p_n of the corresponding sample

contain the number of appearances directly (e.g., anonymised packet traces) or allow it to be estimated in an unbiased manner (e.g., the intrusion alerts discussed in Sect. 2).

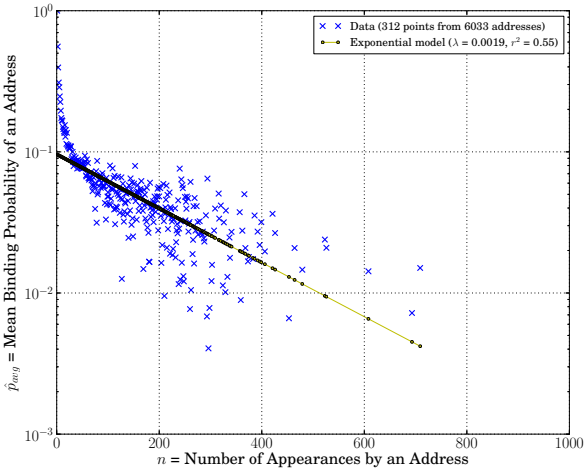


Fig. 8. Exponential mapping between n and the mean p_n of the corresponding sample

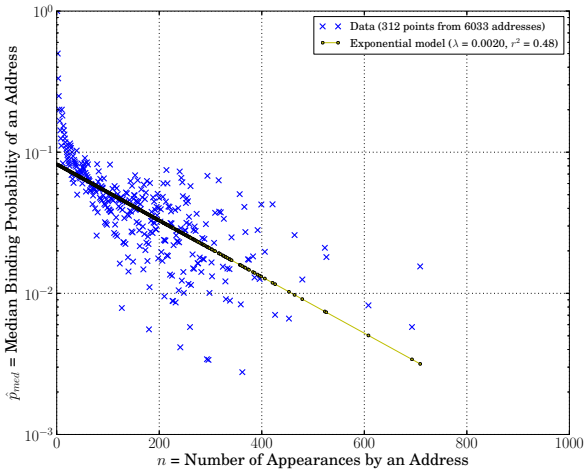


Fig. 9. Exponential mapping between n and median p_n of the corresponding sample

6 Related Work

Our proposed discriminative model is a variant of the two-state Markov Modulated Poisson Process (MMPP) that is well documented in the literature (refer to [7] for an extensive survey). Our focus on the fractal phenomenon is closer to a special case of the MMPP known as a Pareto Modulated Poisson Process

(PMPP) [13]. These processes are primarily concerned with the clustering of bursty traffic into alternating series of ON and OFF periods. In the case of MMPP, these periods are exponentially distributed with a finite variance whereas PMPP allows for power-law distributions with infinite variance in order to capture the emergent statistical self-similarity. We, on the other hand, are concerned with counting the preferential attachment of a latent variable that can take place during state transitions in both of these processes regardless of how the sojourn times in each state are actually distributed. In that respect, our model can be thought of as a special case of the classic Hidden Markov Model (HMM) [20]. However, as we have pointed out at the onset, ours is in the discriminative category while HMMs belong in the generative category. The former is narrow and restrictive due to the computation of a conditional posterior distribution as opposed to a full joint distribution. We must necessarily use a discriminative model since a generative model by definition does not permit anonymity protection given its complete coverage over all the variables and parameters (both hidden and visible).

Various applications of *preferential attachment* (as defined by Barabasi et al. [2]) in the context of computer networks have been well documented in the literature over the years. Refer to [16] for an extensive survey. In the context of dynamic address allocation and volatility, there is a growing body of topical measurements that highlight the manifest effects of preferential attachment between hosts and addresses. Chief among them is the presence of Network Address Translators (NATs) that hide private networks (often large enterprise and campus networks although nowadays more commonly residential broadband routers) [3]. Refer to [4,8,12,15,30] for various detailed measurements of address volatility and the resulting uncertainty. Furthermore, Bellovin [3] proposes a method of counting hosts hidden behind NATs by using the "Identification" field of IPv4 packet headers, which is sometimes implemented as an incrementing counter. However, this method is sensitive to OS variations and is likely to only work with a specific NAT as opposed to all NATs in general.

One recent use of preferential attachment for estimating the host population corresponding to a dynamic address is HostTracker [29], which makes use of application-level user events, e.g., webmail logins, in order to infer a tight binding between hosts and addresses for sufficiently long enough that the bipartite graph mapping hosts to addresses can be derived accurately. However, this was not the trade-off that we were after since the use of privacy sensitive information like email addresses makes it difficult to openly test and validate. Our reliance only on publicly obtainable data provides more transparency and utility with respect to a broad range of network measurement scenarios.

7 Conclusions

In this paper, we have proposed a discriminative model of host to address binding based on preferential attachment at the level of individual addresses. We have also developed efficient and accurate approximations of this model that permit

reliable inferences to be made using real-world data. Our validation focused on the use of anonymised DHCP logs collected at a campus wireless network over the course of four months. We have demonstrated that the number of hosts corresponding to an address is well modelled by the number of times that the address appears. Moreover, this relationship can be expressed as simple mappings between the probability of binding to a new host and the n^{th} appearance by an address. Our work demonstrates that challenging practical scenarios in the context of network management, security and measurement can be handled while still satisfying the legal requirement of anonymity protection.

References

1. Anderson, D.G., Balakrishnan, H., Feamster, N., Koponen, T., Moon, D., Shenker, S.: Accountable Internet Protocol (AIP). In: Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, pp. 339–350. ACM (August 2008)
2. Barabasi, A.L., Albert, R.: Emergence of Scaling in Random Networks. Science 286(5439), 509–512 (1999)
3. Bellovin, S.M.: A Technique for Counting NATted Hosts. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet Measurment, pp. 267–272. ACM, New York (2002)
4. Cai, X., Heidemann, J.: Understanding Block-level Address Usage in the Visible Internet. In: Proceedings of the ACM SIGCOMM Conference, pp. 99–110. ACM (August 2010)
5. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law Distributions in Empirical Data. SIAM Review 51(4), 661–703 (2009)
6. Droms, R.: RFC 2131: Dynamic Host Configuration Protocol. (March 1997), `http://tools.ietf.org/html/rfc2131` (accessed on January 25, 2010)
7. Fischer, W., Meier-Hellstern, K.: The Markov-modulated Poisson process (MMPP) cookbook. Performance Evaluation 18, 149–171 (1992)
8. Heidemann, J., Pradkin, Y., Govindan, R., Papadopoulos, C., Bartlett, G., Bannister, J.: Census and Survey of the Visible Internet. In: Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement, pp. 169–182. ACM (October 2008)
9. Holz, T., Gorecki, C., Reick, K., Freiling, F.C.: Measuring and Detecting Fast-Flux Service Networks. In: Proceedings of the 16th Annual Network & Distributed System Security Symposium, ISOC (February 2008)
10. Information Sciences Institute, University of Southern California: RFC 791: Internet Protocol (September 1981), `http://tools.ietf.org/html/rfc791` (accessed on January 25, 2010)
11. Khadilkar, M., Feamster, N., Sanders, M., Clark, R.: Usage-Based DHCP Lease Time Optimization. In: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement. ACM (2007)
12. Kreibich, C., Weaver, N., Nechaev, B., Paxson, V.: Netalyzr: Illuminating The Edge Network. In: Proceedings of ACM Internet Measurement Conference. ACM (November 2010)
13. Le-Ngoc, T., Subramanian, S.: A Pareto-modulated Poisson process (PMPP) model for long-range dependent traffic. Computer Communications 23, 123–132 (2000)

14. Leland, W.E., Taqqu, M.S., Willinger, W., Wilson, D.: On the Self-Similar Nature of Ethernet Traffic (Extended Version). IEEE/ACM Transactions on Networking 2(1), 1–15 (1994)
15. Maier, G., Feldmann, A., Paxson, V., Allman, M.: On Dominant Characteristics of Residential Broadband Internet Traffic. In: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement, pp. 90–102. ACM (November 2009)
16. Mitzenmacher, M.: A Brief History of Generative Models for Power Law and Lognormal Distributions. Internet Mathematics 1(2), 226–251 (2004)
17. Mockapetris, P.: RFC 1035: Domain Names: Implementation and Specification (November 1987), http://tools.ietf.org/html/rfc1035 (accessed on January 25, 2010)
18. Osterweil, E., Amante, S., McPherson, D., Massey, D.: The Great IPv4 Land Grab: Resource Certification for the IPv4 Grey Market. In: Proceedings of the Tenth ACM Workshop on Hot Topics in Networks. ACM (November 2011)
19. Paxson, V., Floyd, S.: Wide-Area Traffic: The Failure of Poisson Modeling. IEEE/ACM Transactions on Networking 3(3), 226–244 (1995)
20. Rabiner, L.R., Juang, B.H.: An Introduction to Hidden Markov Models. IEEE ASSP Magazine 3(1), 4–16 (1986)
21. Rajab, M.A., Zarfoss, J., Monrose, F., Terzis, A.: My Botnet is Bigger than Yours (Maybe, Better than Yours): why size estimates remain challenging. In: Proceedings of the 1st Workshop on Hot Topics in Understanding Botnets. USENIX Association (April 2007)
22. Rigney, C., Willens, S., Rubens, A., Simpson, W.: RFC 2865: Remote Authentication Dial In User Service (June 2000), http://tools.ietf.org/html/rfc2865 (accessed on January 25, 2010)
23. Taqqu, M.S., Willinger, W., Sherman, R.: Proof of a Fundamental Result in Self-Similar Traffic Modeling. ACM SIGCOMM Computer Communications Review 27(2), 5–23 (1997)
24. Tsuchiya, P.F., Eng, T.: Extending the IP Internet Through Address Reuse. ACM SIGCOMM Computer Communication Review 23(1), 16–33 (1993)
25. Victorian Consolidated Legislation: Information Privacy Act 2000 - SCHEDULE 1 (2000), http://goo.gl/PAFdZ (accessed on June 12, 2012)
26. Hsu, W., Helmy, A.: CRAWDAD trace set usc/mobilib/dhcp (v. 2007-01-08) (January 2007), http://crawdad.cs.dartmouth.edu/usc/mobilib/dhcp (accessed on March 5, 2011)
27. Wilcox, C., Papadopoulos, C., Heidemann, J.: Correlating Spam Activity with IP Address Characteristics. In: Proceedings of INFOCOM IEEE Conference on Computer Communications. IEEE (2010)
28. Willinger, W., Taqqu, M.S., Sherman, R., Wilson, D.V.: Self-Similarity Through High Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level. IEEE/ACM Transactions on Networking 5(1), 71–86 (1997)
29. Xie, Y., Yu, F., Abadi, M.: De-anonymizing the Internet Using Unreliable IDs. In: Proceedings of the 2009 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp. 75–86. ACM (August 2009)
30. Xie, Y., Yu, F., Achan, K., Gillum, E., Goldszmidt, M., Wobber, T.: How Dynamic are IP Addresses? In: Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, pp. 301–312. ACM (August 2007)

Efficient and Robust Identity-Based Handoff Authentication in Wireless Networks

Qi Han, Yinghui Zhang, Xiaofeng Chen*,**, Hui Li, and Jiaxiang Quan

State Key Laboratory of Integrated Service Networks (ISN),
Xidian University, Xi'an 710071, P.R. China
xfchen@xidian.edu.cn

Abstract. In this paper, we propose a new identity-based construction for secure and efficient handoff authentication schemes, in which a special double-trapdoor chameleon hash function serves as the primary ingredient. Compared with all the existing identity-based handoff-related schemes, the main advantage of the proposed scheme eliminates the assumption that PKG is fully trusted. Besides, we show that the proposed scheme not only provides robust security properties, but also enjoys desirable efficiency for the real-world applications.

Keywords: Handoff authentication, Identity-based cryptography, Wireless network.

1 Introduction

With the rapid development of wireless technologies, various wireless networks have found the widespread acceptance and implementation. Due to the convenience and mobility of mobile devices, the handoff request becomes more frequent, which initiates the demand for a secure and efficient handoff authentication scheme. In the past decade, plenty of handoff authentication schemes have been proposed in the literatures [1–8, 10–12].

In IEEE 802.11i, the handoff authentication takes more than 200 *ms* [1], which is unacceptable for real-time traffic. For fast handoff authentication, Mishra et al. [2] proposed a proactive key distribution approach, in which neighbor graph knowledge is utilized to distribute a new pairwise master key (PMK) to neighboring access points (APs). It can reduce the handoff authentication time to 21 *ms* on average, but at the expense of increasing the burden on the authentication, authorization, and accounting (AAA) server and losing the scalability. The above scheme is called the AAA-based scheme and has also been studied in [3, 4]. An alternative approach without communicating with the AAA server is the Security Context Transfer (SCT) scheme. Wang and Prasad [5] proposed a

* Supported by the National Natural Science Foundation of China (Nos.60970144 and 61102056), China 111 Project (No.B08038), and the Fundamental Research Funds for the Central Universities (Nos.K50510010003, JY10000901034 and K50511010001).

** Corresponding author.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 180–191, 2012.

fast SCT scheme by exchanging a random nonce. Specifically, this approach has also been studied in [6, 7]. Although having the advantage of needing no communication between an AAA server and an AP, these SCT schemes still subject to the establishment of a trust relationship among APs, which makes them lose the practicability allowing for the overall system complexity.

Different from the AAA-based schemes and the SCT schemes, Choi and Jung [8] have proposed a handoff authentication scheme using a credential based on chameleon hashing [9]. The scheme generates an authenticated Pairwise Transient Key (PTK) between a MN and a new AP without any prearrangements. Unfortunately, the scheme [8] has the disadvantages of heavy communication and large storage. Morever, Yoon et al. [10] showed that the scheme [8] still fails to achieve PFS/PBS, *i.e.*, the session key PMK will be achieved by any adversaries who have compromised the *long-term* secret key of the MN or AP. In addition, Kim et al. [11] proposed an identity-based handoff authentication scheme. Very recently, Zhang et al. [12] also proposed a new identity-based handoff authentication scheme.

All of the existing identity-based handoff authentication schemes are only secure under the assumption that PKG is fully trustworthy. To the best of our knowledge, it seems that no identity-based handoff-related schemes using untrusted PKG can simultaneously satisfy robust security properties and enjoy desirable efficiency.

Our Contribution. In this paper, we develop a new identity-based construction for secure and efficient handoff authentication schemes, in which the double-trapdoor chameleon hash function [13] serves as a primary building block. Compared with the existing identity-based handoff authentication schemes [11, 12], our construction can resist various malicious attacks even though PKG is not trustworthy, which has never been achieved in any existing identity-based handoff-related schemes. Therefore, our construction is more suitable for handoff authentication in the wireless applications environment.

1.1 Related Work

The notion of identity-based cryptography was introduced by Shamir [14]. Unlike conventional public key cryptography where the recipient's public key must be provided to the sender before starting communication, known identity information is used directly to derive a public key in the identity-based setting. The private keys corresponding to the public keys are generated from a master secret by a trusted authority called Private Key Generator (PKG).

Chameleon hash functions, firstly introduced by Krawczyk and Rabin [9], are a kind of trapdoor one-way hash functions. Chameleon hash functions were originally used to design chameleon signatures [9], which provide non-repudiation and non-transferability for the signed message simultaneously as undeniable signatures [15] do, but the former allows for more efficient realization than the latter. In chameleon signature schemes, the recipient is the holder of trapdoor information. As one limitation of the original chameleon signature schemes, the

key exposure problem of chameleon hashing was firstly addressed by Ateniese and de Medeiros [16]. Chen et al. [17] proposed the first full construction of a key-exposure free chameleon hash function. Later, Ateniese and de Medeiros presented several constructions of key-exposure free chameleon hash functions based on different cryptographic assumptions [18]. On the other hand, Shamir and Tauman [19] first used the chameleon hash functions to develop a new paradigm, named "hash-sign-switch", for designing much more efficient on-line/off-line signatures, which were introduced by Even, Goldreich and Micali [20, 21]. In an on-line/off-line signature scheme, the signer is the holder of the trapdoor information [22]. However, it seems that all existing on-line/off-line signature schemes based on Shamir-Tauman's paradigm still suffer from the key exposure problem of chameleon hashing. Chen et al. [13] first addressed this problem by introducing a double-trapdoor hash family based on the discrete logarithm assumption and then applied the "hash-sign-switch" paradigm to propose a much more efficient generic on-line/off-line signature scheme without key exposure. Recently, Harn et al. [23] proposed some constructions for the double-trapdoor hash family (which they named multiple-collision trapdoor hash family) based on discrete logarithm and factoring assumptions.

1.2 Organization

The rest of the paper is organized as follows: Some preliminaries are given in Section 2. The identity-based construction for handoff authentication schemes is presented in Section 3. The performance related issues and some security considerations are discussed in Section 4. Finally, Section 5 concludes the paper.

2 Preliminaries

In this section, we first introduce a special double-trapdoor chameleon hash function. Then, the basic requirements of a handoff authentication scheme are listed.

2.1 Definition

Definition 1. *(A Double-trapdoor Chameleon Hash Family)[13]*

- **System Parameters Generation:** Let t be a prime power, and $E(\mathbb{F}_t)$ an elliptic curve over finite field $\mathbb{F}_t$. Let $\#E(\mathbb{F}_t)$ be the number of points of $E(\mathbb{F}_t)$, and P be a point of $E(\mathbb{F}_t)$ with prime order q, where $q|\#E(\mathbb{F}_t)$. Denote by $\mathbb{G}$ the subgroup generated by P. Define a cryptographic secure hash function $f\colon \mathbb{Z}_q \times \mathbb{G} \to \mathbb{Z}_q$. Choose two random elements $k, x \in_R \mathbb{Z}_q^*$, and compute $K = kP$, $Y = xP$. The public hash key is $HK = (K, Y)$, and the private trapdoor key is $TK = (k, x)$.
- **The Hash Family:** Given the hash key HK, the double-trapdoor chameleon hash function $H_{HK}\colon \mathbb{Z}_q \times \mathbb{Z}_q \to \mathbb{G}$ is defined as $H_{HK}(m, r) = f(m, K) \cdot K + rY$.

The described double-trapdoor chameleon hash function satisfies the following properties:

1. **Efficiency:** Given the hash key HK and a pair $(m, r) \in \mathbb{Z}_q \times \mathbb{Z}_q$, $H_{HK}(m, r) = f(m, K) \cdot K + rY$ is computable in polynomial time.
2. **Collision Resistance:** Without the trapdoor key TK, it is computationally infeasible to find two pairs $(m_1, r_1), (m_2, r_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$ that satisfy $m_1 \neq m_2$ and $H_{HK}(m_1, r_1) = H_{HK}(m_2, r_2)$.
3. **Trapdoor Collisions:** Assume that we are given the hash and trapdoor key pair (HK, TK), a pair $(m_1, r_1) \in \mathbb{Z}_q \times \mathbb{Z}_q$, and an additional message $m_2 \in \mathbb{Z}_q$, we want to find $r_2 \in \mathbb{Z}_q$ such that $f(m_1, kP) \cdot kP + r_1 Y = f(m_2, kP) \cdot kP + r_2 Y$. The value of r_2 can be computed in polynomial time as follows: $r_2 = r_1 + kx^{-1}(f(m_1, kP) - f(m_2, kP)) \mod q$.

2.2 Requirements of Handoff Authentication

A handoff authentication scheme in wireless networks should satisfy the following requirements.

- **Mutual Authentication:** During a process of handoff authentication, a basic objective is realizing the mutual authentication between a MN and an AP. On the one hand, the MN is required to be authenticated by the AP as a legitimate mobile node. On the other hand, to prevent attackers from compromising an AP, it is necessary for the MN to authenticate the AP.
- **Key Agreement:** A practical handoff authentication scheme should allow a MN and an AP to establish a PTK as a shared session key, which can ensure the confidentiality of the subsequent communications between the MN and the AP.
- **Robust Security Property:** For one thing, a secure handoff authentication scheme should provide robust security against various kinds of attacks including passive eavesdropping, impersonation attack, replay attack, man-in-the-middle attack, *etc.* For another, the scheme should possess some significant security benefits such as resistance against untrusted PKG, the perfect forward/backward secrecy (PFS/PBS).
- **Desirable Efficiency:** Considering the fact that mobile nodes in wireless networks are resource-constrained and wireless channels are bandwidth constrained and error-prone, the design of a handoff authentication scheme for the wireless environment should enjoy desirable efficiency.

3 The Proposed Handoff Authentication Scheme

In this section, we present an identity-based handoff authentication scheme. The proposed scheme consists of two primary processes of the initial full authentication and the handoff authentication. In the following, we first set up system parameters, and then the detailed processes of the developed handoff authentication scheme are described.

3.1 System Parameters Generation

Let t be a prime power, and $E(\mathbb{F}_t)$ an elliptic curve over finite field $\mathbb{F}_t$. Let $\#E(\mathbb{F}_t)$ be the number of points of $E(\mathbb{F}_t)$, and P be a point of $E(\mathbb{F}_t)$ with prime order q, where $q|\#E(\mathbb{F}_t)$. Denote by $\mathbb{G}$ the additive cyclic group generated by P and $\mathbb{G}_T$ the multiplicative cyclic group of order q. Let $e\colon \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear map. PKG randomly chooses $s \in_R \mathbb{Z}_q^*$. Define three cryptographic secure hash functions $H\colon \{0,1\}^* \times \mathbb{G} \to \mathbb{G}$, $f_1\colon \mathbb{Z}_q^* \times \mathbb{G} \times \{0,1\}^{l_1} \to \mathbb{Z}_q^*$ and $f_2\colon \mathbb{G} \times \mathbb{G} \to \{0,1\}^{l_2}$, where l_1 denotes the binary length of related time compoments such as T_{Curr} and T_{Exp}, and $l_2 = 160$ when f_2 is designated as SHA-1. Given a hash key $HK = (K, Y)$, the double-trapdoor chameleon hash function $H_{HK}\colon \mathbb{Z}_q^* \times \mathbb{Z}_q^* \to \mathbb{G}$ is defined as follows: $H_{HK}(m, r) = f_1(m \parallel K \parallel T_{\text{Curr}}) \cdot K + rY$. The system parameters are $SysParas = \{E, t, q, P, e, \mathbb{G}, \mathbb{G}_T, H, f_1, f_2, H_{HK}\}$, which are held by all nodes.

3.2 Process of Initial Full Authentication

Two phases, network authentication phase and handoff initialization phase, are involved in the process of initial full authentication, which proceeds as follows.

- **Network Authentication Phase:** The network authentication phase can be performed by EAP-TLS, which is similar to the IEEE 802.1x authentication process. Once the authentication succeeds, the MN and AP_1 share a PMK, which can ensure the security of communications in the subsequent handoff initialization phase.
- **Handoff Initialization Phase:** Based on a successful network authentication, the MN can receive a short-term certificate $Cert_{\text{MN}}$ from the AAA server after every handoff initialization phase, which is a preparation for the process of handoff authentication. It is worth noting that the AAA server will issue $Cert_{\text{MN}}$ to APs after every expiration time of the certificate. The detailed steps of the handoff initialization phase are described as follows:
  1. **Key Request Preparation:** The MN chooses at random $x_{\text{MN}} \in_R \mathbb{Z}_q^*$ and compute $Y_{\text{MN}} = x_{\text{MN}}P$. As a result, the key request message is $Message_{\text{KeyRq}} = ID_{\text{MN}} \parallel Y_{\text{MN}}$, where ($x_{\text{MN}}$, Y_{MN}) is the *long-term* trapdoor/hash key pair.
  2. **Key Request:** MN $\longrightarrow$ PKG: $Message_{\text{KeyRq}}$.
  3. **Private Key Generation:** Upon receiving the key request message $Message_{\text{KeyRq}}$ from the MN, PKG computes private keys as follows:

$$SK^*_{\text{MN}} = sH(ID_{\text{MN}} \parallel Y_{\text{MN}} \parallel T_{\text{Exp}}),$$
$$SK_{\text{MN}} = sH(ID_{\text{MN}} \parallel T_{\text{Exp}}).$$

     Note that T_{Exp} represents the expiration time of private keys. Then PKG sets $Message_{\text{KeyRs}} = SK^*_{\text{MN}} \parallel SK_{\text{MN}} \parallel T_{\text{Exp}}$. Similarly, the AAA server can also obtain the corresponding private keys SK^*_{AAA} and SK_{AAA}.
  4. **Key Response:** PKG $\longrightarrow$ MN: $Message_{\text{KeyRs}}$.

5. **Secret Parameters Generation:** Firstly, the MN chooses at random $k^*_{\mathrm{MN}} \in_R \mathbb{Z}^*_q$ and lets secret parameters $SecParas_{\mathrm{MN}}=(x_{\mathrm{MN}}, k^*_{\mathrm{MN}})$. Secondly, the MN computes x^{-1}_{MN}, and $h_{\mathrm{MN}} = k^*_{\mathrm{MN}} Y_{\mathrm{MN}}$. Finally, the MN computes $V_{\mathrm{MN}} = e(SK^*_{\mathrm{MN}}, ID_{\mathrm{AAA}})$ and sets h_{MN} as a value of the double-trapdoor chameleon hash function H_{HK}. As a result, the certificate request message is

$$\begin{aligned} Message_{\mathrm{CertRq}} &= ID_{\mathrm{MN}} \parallel Y_{\mathrm{MN}} \parallel h_{\mathrm{MN}} \parallel T_{\mathrm{Exp}} \\ &= ID_{\mathrm{MN}} \parallel x_{\mathrm{MN}} P \parallel (k^*_{\mathrm{MN}} \cdot x_{\mathrm{MN}}) P \parallel T_{\mathrm{Exp}}. \end{aligned}$$

6. **Certificate Request:** MN $\longrightarrow$ AAA server: $Message_{\mathrm{CertRq}} \parallel V_{\mathrm{MN}}$.
7. **Certificate Generation:** Upon receiving the certificate request message $Message_{\mathrm{CertRq}}$ and V_{MN} from the MN, the AAA server checks if

$$V_{\mathrm{MN}} = e(H(ID_{\mathrm{MN}} \parallel Y_{\mathrm{MN}} \parallel T_{\mathrm{Exp}}), SK_{\mathrm{AAA}}).$$

   If it is, the MN computes $Cert_{\mathrm{MN}} = Sign(SK_{\mathrm{AAA}}, Message_{\mathrm{CertRq}} \parallel T_{\mathrm{Exp}})$, where *Sign* is any provably secure identity-based signature algorithm, such as the Hess signature [24]. Then the certificate response message is $Message_{\mathrm{CertRs}} = Cert_{\mathrm{MN}} \parallel T_{\mathrm{Exp}}$.
8. **Certificate Response:** the AAA server $\longrightarrow$ MN: $Message_{\mathrm{CertRs}}$.
9. **Handoff Parameters Setup:** After receiving the certificate response message $Message_{\mathrm{CertRs}}= Cert_{\mathrm{MN}} \parallel T_{\mathrm{Exp}}$ from the AAA server, the MN sets $SecParas = (x_{\mathrm{MN}}, k^*_{\mathrm{MN}})$ as its secret handoff parameters and stores $Cert_{\mathrm{MN}}$, which are necessary in the process of handoff authentication.

Remark 1. By performing the handoff initialization phase, AP_2 sets up its own handoff parameters $SecParas_{\mathrm{AP}_2}=(x_{\mathrm{AP}_2}, k^*_{\mathrm{AP}_2})$. Specially, the *long-term* trapdoor/hash key pair of AP_2 is $(x_{\mathrm{AP}_2}, Y_{\mathrm{AP}_2})$, *i.e.*, $(x_{\mathrm{AP}_2}, x^*_{\mathrm{AP}_2} P)$.

3.3 Process of Handoff Authentication

1. **Trapdoor Collision:** The MN chooses at random $k_{\mathrm{MN}}, m_{\mathrm{MN}} \in_R \mathbb{Z}^*_q$ and finds a new trapdoor collision as follows:

$$r_{\mathrm{MN}} = k^*_{\mathrm{MN}} - f_1(m_{\mathrm{MN}} \parallel k_{\mathrm{MN}} P \parallel T_{\mathrm{Curr}}) \cdot k_{\mathrm{MN}} \cdot x^{-1}_{\mathrm{MN}}.$$

   Note that T_{Curr} represents a timestamp. In addition, the MN deduces verification information: $VerifyInfo_{\mathrm{MN}} = m_{\mathrm{MN}} \parallel k_{\mathrm{MN}} P \parallel r_{\mathrm{MN}} Y_{\mathrm{MN}} \parallel Cert_{\mathrm{MN}} \parallel ID_{\mathrm{MN}} \parallel T_{\mathrm{Curr}} \parallel T_{\mathrm{Exp}}$, where $(k_{\mathrm{MN}} P, k_{\mathrm{MN}} \cdot x^{-1}_{\mathrm{MN}})$ is the *one-time* hash/trapdoor key pair.
2. **Send Verification Information:** MN $\longrightarrow$ AP_2: $VerifyInfo_{\mathrm{MN}}$.
3. Upon receiving the verification information $VerifyInfo_{\mathrm{MN}}$, AP_2 can check the identity validity of the MN through the following identity authentication. If valid, the subsequent steps are performed.

- **Identity Authentication:** After receiving $VerifyInfo_{\mathrm{MN}}$ from the MN, AP_2 computes $h'_{\mathrm{MN}} = f_1(m_{\mathrm{MN}} \parallel k_{\mathrm{MN}}P \parallel T_{\mathrm{Curr}}) \cdot k_{\mathrm{MN}}P + r_{\mathrm{MN}}Y_{\mathrm{MN}}$, and then authenticates the MN according to the following Equation (1).

$$Verify(Cert_{\mathrm{MN}}, PK_{\mathrm{AAA}}, ID_{\mathrm{MN}} \parallel Y_{\mathrm{MN}} \parallel h'_{\mathrm{MN}} \parallel T_{\mathrm{Exp}}) = \text{“valid”}, \quad (1)$$

  where $Verify$ represents the verification of the Hess signature [24]. AP_2 regards the MN as a legitimate node if and only if Equation (1) holds.
- **Trapdoor Collision:** If the MN is legitimate, AP_2 chooses at random $k_{\mathrm{AP}_2}, m_{\mathrm{AP}_2} \in_R \mathbb{Z}_q^*$ and finds a new trapdoor collision as follows:

$$r_{\mathrm{AP}_2} = k^*_{\mathrm{AP}_2} - f_1(m_{\mathrm{AP}_2} \parallel k_{\mathrm{AP}_2}P \parallel T_{\mathrm{Curr}}) \cdot k_{\mathrm{AP}_2} \cdot x^{-1}_{\mathrm{AP}_2}.$$

  In addition, AP_2 deduces verification information: $VerifyInfo_{\mathrm{AP}_2} = m_{\mathrm{AP}_2} \parallel k_{\mathrm{AP}_2}P \parallel r_{\mathrm{AP}_2}Y_{\mathrm{AP}_2} \parallel Cert_{\mathrm{AP}_2} \parallel ID_{\mathrm{AP}_2} \parallel T_{\mathrm{Curr}} \parallel T_{\mathrm{Exp}}$.
- **Key Agreement:** AP_2 computes $\mathrm{PTK} = k_{\mathrm{AP}_2} \cdot (k_{\mathrm{MN}}P)$ and deduces confirm information $ConfirmInfo_{\mathrm{AP}_2} = f_2(\mathrm{PTK} \parallel k_{\mathrm{MN}}P)$.

4. **Send Verify-Confirm Information:** $\mathrm{AP}_2 \longrightarrow$ MN: $VerifyInfo_{\mathrm{AP}_2} \parallel ConfirmInfo_{\mathrm{AP}_2}$.
5. Upon receiving the verify-confirm information $VerifyInfo_{\mathrm{AP}_2} \parallel ConfirmInfo_{\mathrm{AP}_2}$ from AP_2, the MN checks the identity validity of AP_2 via the following identity authentication. If valid, the following steps, key agreement and PTK confirmation, are performed.
   - **Identity Authentication:** After receiving from AP_2 the verify-confirm information $VerifyInfo_{\mathrm{AP}_2} \parallel ConfirmInfo_{\mathrm{AP}_2}$, the MN computes

$$h'_{\mathrm{AP}_2} = f_1(m_{\mathrm{AP}_2} \parallel k_{\mathrm{AP}_2}P \parallel T_{\mathrm{Curr}}) \cdot k_{\mathrm{AP}_2}P + r_{\mathrm{AP}_2}Y_{\mathrm{AP}_2},$$

     and then authenticates AP_2 according to Equation (2).

$$Verify(Cert_{\mathrm{AP}_2}, PK_{\mathrm{AAA}}, ID_{\mathrm{AP}_2} \parallel Y_{\mathrm{AP}_2} \parallel h'_{\mathrm{AP}_2} \parallel T_{\mathrm{Exp}}) = \text{“valid”}, \quad (2)$$

     where $Verify$ represents the verification of the Hess signature [24]. The MN regards AP_2 as a legitimate AP if and only if Equation (2) holds.
   - **Key Agreement:** If AP_2 is legitimate, the MN computes $\mathrm{PTK} = k_{\mathrm{MN}} \cdot (k_{\mathrm{AP}_2}P)$.
   - **PTK Confirmation:** Using the $ConfirmInfo_{\mathrm{AP}_2}$ from AP_2, the MN thinks that AP_2 has successfully achieved PTK if and only if $f_2(\mathrm{PTK} \parallel k_{\mathrm{MN}}P) = ConfirmInfo_{\mathrm{AP}_2}$. If successful, the MN computes $ConfirmInfo_{\mathrm{MN}} = f_2(\mathrm{PTK} \parallel k_{\mathrm{AP}_2}P)$.
6. **Confirm Response:** After ensuring that AP_2 has successfully obtained PTK, MN $\longrightarrow \mathrm{AP}_2$: $ConfirmInfo_{\mathrm{MN}}$.
7. **PTK Confirmation:** Upon receiving the confirm information $ConfirmInfo_{\mathrm{MN}}$, AP_2 thinks that the MN has successfully achieved PTK if and only if $f_2(\mathrm{PTK} \parallel k_{\mathrm{AP}_2}P) = ConfirmInfo_{\mathrm{MN}}$.

After all the above procedures in the process of handoff authentication are performed successfully, the MN and AP_2 can accomplish mutual authentication with key agreement. The shared key $PTK = (k_{MN} \cdot k_{AP_2})P$ can be used to protect the subsequent communications data between the MN and AP_2 over the air interface. Obviously, it concludes that $h'_n = k^*_n Y_n$. Therefore, if both the MN and AP_2 follow the procedures of our construction, it can be guaranteed that the identity authentication is valid.

4 Analysis of the Proposed Scheme

4.1 Security Analysis

We prove that the proposed scheme provides robust security properties.

- **Mutual Authentication:** Our construction ensures only the legitimate MN to access the wireless networks successfully. To be specific, the proposed scheme deduces a value $h_n = (k^*_n \cdot x_n)P$ from secret keys $SecParas_n$ and uses it as a value of the double-trapdoor chameleon hash function H_{HK}. And then certificate $Cert_n$ is generated by signing h_n with the signature key of the AAA server. The MN and AP_2 can authenticate each other by checking the validity of certificates $Cert_{AP_2}$ and $Cert_{MN}$, respectively. According to equation $r_n = k^*_n - f_1(m_n \parallel k_n P \parallel T_{Curr}) \cdot k_n \cdot x_n^{-1}$, an adversary who does not know the MN's or AP's secret keys $SecParas_n = (x_n, k^*_n)$ cannot find valid trapdoor collision, and thus cannot obtain valid authentication messages. Accordingly, our construction can realize the mutual authentication between the MN and AP_2.
- **Key Agreement:** By performing the procedures of our construction, a pairwise transient key PTK, as a future shared session key, can be agreed upon between the MN and AP_2 based on the elliptic curve Diffie Hellman (ECDH) key exchange protocol. Specifically speaking, our scheme allows each node to choose at random $k_n \in_R \mathbb{Z}^*_q$ as ECDH secret key without contacting the AAA server or establishing trust relationship between APs. After successful identity authentication, k_n can be used to establish a shared session key. In our construction, PTK can be shared by the MN and AP_2, which satisfies equation $PTK = (k_{MN} \cdot k_{AP_2})P$.
- **Passive Eavesdropping:** During the process of the handoff authentication, all the messages available for eavesdroppers derive from identity verification information $VerifyInfo_n$ and PTK confirm information $ConfirmInfo_n$. Under the computational intractability assumption of the elliptic curve discrete logarithm problem (ECDLP) [25], it is obvious that these information will not leak both the secret information of the MN and AP_2. Moreover, after the handoff authentication, the communication data is encrypted using PTK. Therefore, our construction can resist against passive eavesdropping.
- **Impersonation Attack:** A bogus MN may pick up a MAC address from a legitimate MN. Subsequently, it impersonates the legitimate MN by modifying its own MAC address and tries the process of handoff authentication.

However, it cannot perform the critical key agreement of Step 5 in the process of handoff authentication due to the lack of secret keys. For the same reason, a bogus AP cannot perform the key agreement. So our construction can resist against the impersonation attack.

- **Replay Attack:** In the process of handoff authentication, an adversary may record the message of Step 2 and replay it. Our construction of the handoff authentication scheme can prevent from the replay of previous messages using the timestamp. Since there is no sequence number, our scheme cannot detect a replay of the message of Step 2 during the current process of handoff authentication. Nevertheless, the adversary still cannot access an AP because of lacking the valid secret keys and thus failing to pass Step 3. Accordingly, it is worthless for an adversary to replay messages.
- **Man-in-the-Middle Attack:** An active adversary in the middle of the communications between the MN and AP_2 can intercept and block the MN's verification information $VerifyInfo_{\mathrm{MN}}$. For his own interests, the adversary may send $VerifyInfo_{\mathrm{ad}}$ to AP_2, which is obtained by embedding information relevant with the adversary into $VerifyInfo_{\mathrm{MN}}$. However, based on the security of the signature algorithm adopted by the AAA server, he cannot pass the identity authentication of Step 3 in the process of handoff authentication for the lack of secret keys. Certainly, the adversary cannot masquerade as the MN and agree another PTK with AP_2. Hence, our construction can resist against the man-in-the-middle attack.
- **Untrusted PKG:** In the existing identity-based schemes [11, 12], the handoff authentication is based on the private key of MNs and APs, so PKG is fully trusted owing to the problem of key escrow in the identity-based cryptosystem, *i.e.*, PKG knows the private keys of all users. In our construction, due to the lack of secret handoff parameter x_{MN}, PKG cannot compute the trapdoor collision r_{MN}. Additionally, after the network authentication phase is fulfilled by EAP-TLS, the communication security between the MN and the AAA server is ensured in the subsequent handoff initialization phase. Therefore, the certificate request message $Message_{\mathrm{CertRq}}$ is not known to the PKG. As a result, the PKG still cannot forge certificates of any mobile node although it has the AAA server's secret key. Hence it is impossible for PKG to pretend to be any mobile node. Similarly, PKG cannot pretend to be APs. It is thus clear that the proposed scheme is secure even if PKG is not fully trusted.
- **Perfect Forward/Backward Secrecy (PFS/PBS):** In our scheme, supposing that the secret keys $SecParas_{\mathrm{n}}=(x_{\mathrm{n}}, k_{\mathrm{n}}^*)$ are known to an attacker at some point in time, then all the information resources available for the attacker consist of $x_{\mathrm{n}}, k_{\mathrm{n}}^*$, and $VerifyInfo_{\mathrm{n}}$, where $VerifyInfo_{\mathrm{n}} = m_{\mathrm{n}} \parallel k_{\mathrm{n}}P \parallel r_{\mathrm{n}}Y_{\mathrm{n}} \parallel Cert_{\mathrm{n}} \parallel ID_{\mathrm{n}} \parallel T_{\mathrm{Curr}} \parallel T_{\mathrm{Exp}}$. Note that ID_{n} is the identity of a mobile node n. In order to achieve the previous or the following session keys PTKs, it is necessary for the attacker to compute k_{n} from the following trapdoor collision equation.

$$r_{\mathrm{n}} = k_{\mathrm{n}}^* - f_1(m_{\mathrm{n}} \parallel k_{\mathrm{n}}P \parallel T_{\mathrm{Curr}}) \cdot k_{\mathrm{n}} \cdot x_{\mathrm{n}}^{-1}. \tag{3}$$

However, we can show that k_{n} still cannot be obtained by the attacker. We consider the following cases:

(1) The attacker tries to compute k_{n} by directly solving Equation (3). Considering that Equation (3) is a indefinite equation with r_{n} and k_{n} as variables, the attacker cannot find k_{n}. In addition, under the intractability assumption of ECDLP, it is infeasible for the attacker to compute r_{n} using $r_{\mathrm{n}}Y_{\mathrm{n}}$, which can be extracted from $\mathit{VerifyInfo}_{\mathrm{n}}$.

(2) The attacker multiplies P to the two sides of Equation (3) and gets Equation (4).

$$r_{\mathrm{n}}P = k_{\mathrm{n}}^{*}P - f_1(m_{\mathrm{n}} \parallel k_{\mathrm{n}}P \parallel T_{\mathrm{Curr}}) \cdot k_{\mathrm{n}} \cdot x_{\mathrm{n}}^{-1}P. \tag{4}$$

Moreover, the attacker may pick up $r_{\mathrm{n}}P$ from $\mathit{VerifyInfo}_{\mathrm{n}}$. Then he tries to deduce k_{n} from the tuple $((f_1(m_{\mathrm{n}} \parallel k_{\mathrm{n}}P \parallel T_{\mathrm{Curr}}) \cdot x_{\mathrm{n}}^{-1}P),\ k_{\mathrm{n}}^{*}P - r_{\mathrm{n}}P)$, which is, however, infeasible under the intractability assumption of ECDLP.

4.2 Performance Analysis

We evaluate the computation cost of the proposed scheme. We only consider the time-consuming operations such as pairing, exponentiation and scalar multiplication etc., and omit other operations such as modular addition, modular multiplication and hash in all three schemes. Let $Pair$ be a bilinear pairing operation, Exp an exponentiation in $\mathbb{G}_T$ and M a scalar multiplication in $\mathbb{G}$. Table 1 presents the comparison of computation cost between the existing schemes [11, 12] and ours.

Table 1. Comparison of the Computation Cost

	Scheme [11]	Scheme [12]	Our scheme
MN	$2Pair$+$1M$	$1Pair$+$3M$	$2Pair + 2M + 1Exp$
AP_2	$2Pair$+$1M$	$1Pair$+$3M$	$2Pair + 2M + 1Exp$

Though our scheme requires a (very) little more computation overload compared with the schemes [11, 12], it eliminates the assumption that PKG is fully trusted. Therefore, we argue that our scheme is more suitable for the real-world applications.

5 Conclusion

Secure and efficient handoff authentication schemes have found important role in roaming communications. In this paper, we present a new efficient identity-based handoff authentication scheme. In the proposed scheme, a special double-trapdoor chameleon hash function is the primary ingredient. Our scheme can

simultaneously provide Perfect Forward/Backward Secrecy and complete the seamless handoff of mobile nodes even if PKG is not fully trusted. Compared with the existing identity-based handoff-related schemes, the proposed scheme is more suitable for the seamless handoff of mobile nodes in real-world applications.

References

1. Chaplin, C., Qi, E., Ptasinski, H., Walker, J., Li, S.: 802.11i overview, IEEE.802.11-04/0123r1 (February 2005), http://www.drizzle.com/~aboba/IEEE
2. Mishra, A., Shin, M., Arbaugh, W.: Proactive Key Distribution Using Neighbor Graphs. IEEE Wireless Communication Magazine 11(1), 26–36 (2004)
3. Pack, S., Choi, Y.: Fast Handoff Scheme based on mobility prediction in public wireless LAN systems. IEE Proc.-Commun. 151(5), 489–495 (2004)
4. Hong, K., Jung, S., Wu, S.F.: A Hash-Chain Based Authentication Scheme for Fast Handover in Wireless Network. In: Song, J.-S., Kwon, T., Yung, M. (eds.) WISA 2005. LNCS, vol. 3786, pp. 96–107. Springer, Heidelberg (2006)
5. Wang, H., Prasad, A.R.: Fast Authentication for Inter-domain Handover. In: de Souza, J.N., Dini, P., Lorenz, P. (eds.) ICT 2004. LNCS, vol. 3124, pp. 973–982. Springer, Heidelberg (2004)
6. Choi, J., Jung, S.: A Secure and Efficient Handover Authentication Based on Lightweight Diffie-Hellman on Mobile Node in FMIPv6. IEICE Transactions on Communications E-91B(2), 605–608 (2008)
7. Zhang, C., Lu, R., Ho, P., Chen, A.: A location Privacy Preserving Authentication Scheme in Vehicular Networks. In: WCNC 2008, pp. 2543–2548. IEEE Communications Society, New York (2008)
8. Choi, J., Jung, S.: A Handover Authentication Using Credentials Based on Chameleon Hashing. IEEE Communication Letters 14(1), 54–56 (2010)
9. Krawczyk, H., Rabin, T.: Chameleon Signatures. In: NDSS 2000, pp. 143–154 (2000)
10. Yoon, E., Khan, M., Yoo, K.: Cryptanalysis of a Handover Authentication Scheme Using Credentials Based on Chameleon Hashing. IEICE Transactions on Information and Systems E93-D(12), 3400–3402 (2010)
11. Kim, Y., Ren, W., Jo, J., Yang, M., Jiang, J., Zheng: SFRIC: A Secure Fast Roaming Scheme in Wireless LAN Using ID-based Cryptography. In: ICC 2007, pp. 1570–1575. IEEE Communications Society, New York (2007)
12. Zhang, Y., Chen, X., Li, H., Cao, J.: Secure And Efficient Identity-Based Handoff Authentication Schemes. Security and Communication Networks, February 6 (2012), doi:10.1002/sec.421
13. Chen, X., Zhang, F., Susilo, W., Mu, Y.: Efficient Generic On-Line/Off-Line Signatures Without Key Exposure. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 18–30. Springer, Heidelberg (2007)
14. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
15. Chaum, D., van Antwerpen, H.: Undeniable Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 212–216. Springer, Heidelberg (1990)
16. Ateniese, G., de Medeiros, B.: Identity-Based Chameleon Hash and Applications. In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, pp. 164–180. Springer, Heidelberg (2004)

17. Chen, X., Zhang, F., Kim, K.: Chameleon Hashing Without Key Exposure. In: Zhang, K., Zheng, Y. (eds.) ISC 2004. LNCS, vol. 3225, pp. 87–98. Springer, Heidelberg (2004)
18. Ateniese, G., de Medeiros, B.: On the Key Exposure Problem in Chameleon Hashes. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 165–179. Springer, Heidelberg (2005)
19. Shamir, A., Tauman, Y.: Improved Online/Offline Signature Schemes. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 355–367. Springer, Heidelberg (2001)
20. Even, S., Goldreich, O., Micali, S.: On-Line/Off-Line Digital Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 263–275. Springer, Heidelberg (1990)
21. Even, S., Goldreich, O., Micali, S.: On-line/Off-line Digital Signatures. Journal of Cryptology 9(1), 35–67 (1996)
22. Chen, X., Zhang, F., Tian, H., Wei, B., Susilo, W., Mu, Y., Lee, H., Kim, K.: Efficient Generic On-line/Off-line (Threshold) Signatures Without Key Exposure. Information Sciences 178(21), 4192–4203 (2008)
23. Harn, L., Hsin, W., Lin, C.: Efficient On-line/Off-line Signature Schemes Based on Multiple-Collision Trapdoor Hash Families. The Computer Journal 53(9), 1478–1484 (2010)
24. Hess, F.: Efficient Identity Based Signature Schemes Based on Pairings. In: Nyberg, K., Heys, H. (eds.) SAC 2002. LNCS, vol. 2595, pp. 310–324. Springer, Heidelberg (2003)
25. Johnson, D., Menezes, A., Vanstone, S.: The Elliptic Curve Digital Signature Algorithm (ECDSA). International Journal of Information Security 1(1), 36–63 (2001)
26. Lenstra, A.K., Verheul, E.R.: Selecting Cryptographic Key Sizes. Journal of Cryptology 14(4), 255–293 (2001)

An Improved Authentication Scheme for H.264/SVC and Its Performance Evaluation over Non-stationary Wireless Mobile Networks

Yifan Zhao[1,⋆], Swee-Won Lo[2], Robert H. Deng[2], and Xuhua Ding[2]

[1] College of Computer Science and Technology,
Zhejiang University, Hangzhou, China
yifan0216@gmail.com
[2] School of Information Systems, Singapore Management University, Singapore
{sweewon.lo.2009,robertdeng,xhding}@smu.edu.sg

Abstract. In this paper, a bit stream-based authentication scheme for H.264/Scalable Video Coding (SVC) is proposed. The proposed scheme seamlessly integrates cryptographic algorithms and erasure correction codes (ECCs) to SVC video streams such that the authenticated streams are format compliant with the SVC specifications and preserve the three dimensional scalability (i. e., spatial, quality and temporal) of the original streams. We implement our scheme on a smart phone and study its performance over a realistic bursty packet-lossy wireless mobile network. Our analysis and experimental results show that the scheme achieves very high verification rates with lower communication overhead and much smaller decoding delay compared with the existing solutions.

Keywords: Authentication, data integrity protection, H.264/SVC, Gilbert channel model, multimedia security.

1 Introduction

Digital video is one of the fundamental building blocks in applications such as remote education, telemedicine and IPTV, where video bit streams are commonly delivered to users through wired and wireless networks and are consumed on devices ranging from high-end terminals (e. g. home TVs) to low-power mobile devices (e. g. smart phones). As a result, service providers face the challenge of catering their videos for devices with different capabilities over packet lossy networks with varying bandwidths. A straight-forward solution to the above challenge is to keep multiple versions of a video, each catered for a specific device capability and network bandwidth, and forward the most appropriate version to a user. This approach is inefficient since it requires transmitting and managing multiple versions of every video. The state-of-the-art H.264/SVC (Scalable Video Coding) standard is designed to address the above problem. H.264/SVC

⋆ He was an exchange student to the School of Information Systems, Singapore Management University, during August 2011 to April 2012.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 192–205, 2012.

provides spatial, quality and temporal scalability by encoding a video into a *layered* bit stream. As the bit stream is sent over a source-proxy-user network, one or more proxies in the network perform video downscaling by removing one or more higher layers. This implies that the video source only needs to maintain and transmit a single highest-quality stream per video content that can be adapted by network proxies for differing device capabilities and network bandwidths.

Video streams delivered over public source-proxy-user networks are vulnerable to malicious tampering which may be motivated by commercial or political purposes. In this paper, we are concerned with authentication and integrity protection of H.264/SVC bit streams over such networks. Ideally, an authentication scheme for scalable video should possess the following desirable properties: low processing *delay* at the source and end user devices to support real-time applications, low *computation cost* to be usable by low-power devices, and high authentication verification rates over packet-lossy networks (i. e., *loss-resiliency*) with minimum *communication overhead*. Unlike traditional data authentication techniques which treat any modification on data as tampering, an authentication scheme for scalable video must preserve the scalability of the original video bit stream such that the authenticated bit stream remains verifiable after legitimate downscaling by network proxies. Finally, the scheme should be *proxy-transparent* in the sense that a proxy can perform downscaling without needing to be aware of the underlying security mechanism; this is important in scenarios where a proxy handles multiple video streams from many sources to many end users.

1.1 Related Work

Most existing schemes are designed for non-scalable streams. In [4], a hash chain is applied on packets of a stream and the last packet is signed, but the stream is unverifiable if a packet is lost; [7, 14, 24, 26, 28] then replace the hash chain with different hash graphs, but communication overhead is a trade-off with loss-resiliency and is generally at a multiple of hashes. Erasure correction code (ECC) is used in SAIDA [20] and cSAIDA [19] to lower communication overhead, where a stream is processed in groups of n packets and ECC is applied on the authentication data; as long as k or more packets ($k < n$) are received, the authentication data can be recovered. cSAIDA [19] has a much lower communication overhead compared to SAIDA [20] by performing ECC twice. An excellent survey on authentication schemes for non-scalable streams can be found in [8].

For scalable video streams, three related authentication schemes for MPEG-4 video are proposed in [27]; two of the schemes use a technique similar to hash chaining while the third uses Merkle hash tree (MHT). An MPEG-4 stream is processed in groups of n pictures and authentication data is protected using the method in [19]. Although these schemes have low communication overhead and are loss-resilient, they are not proxy-transparent. Temporal scalability (in which pictures[1] are *selectively* discarded by proxies) is not supported. Similar limitations are observed in [12] and [1]. In [8], a video stream is processed in

[1] We use the terms "picture" and "access unit" interchangeably.

groups of n pictures, each picture has one or more layers. A hash chain is created starting from the highest layer; the hash of a higher layer is appended to its lower layer and the hash of the lowest layer is regarded as the picture's hash. Each picture hash is then appended to its previous picture and the first picture is signed. This scheme is proxy-transparent but is not loss-resilient and does not support temporal scalability. In [5, 11], solutions using hash chain and MHT are proposed but proxy-transparency and loss-resiliency are not addressed.

The work most related to ours is the novel H.264/SVC stream authentication scheme proposed in [15], which, to our knowledge, is the first scheme that preserves all three dimensional scalabilities. In this scheme, the spatial and quality[2] layers are authenticated using a directed acyclic graph and temporal layers are authenticated using a hash chain. Authentication data is protected using ECC and packet replication. The scheme is proxy-transparent and is shown to be resilient to bursty packet losses using packet interleaving, but it incurs a relatively high communication overhead and long decoding delay. In addition, the use of hash chain on temporal layers makes the scheme less robust to packet losses (see Sect. 3.1). There is also a higher source delay since the source's signature is generated over several Groups-of-Pictures (GOPs), which makes it unsuitable for real-time applications.

1.2 Our Contributions

In this paper, an improved bit stream authentication scheme focusing on the H.264/SVC Network Abstraction Layer Units (NALUs) is proposed. Apart from guaranteeing integrity and authenticity, the scheme is capable of supporting all three dimensions of scalability provided by H.264/SVC, satisfies the requirements of real-time applications, and is resilient to packet losses over non-stationary wireless networks. Specifically, our scheme is H.264/SVC format compliant, *scalable* and *proxy-transparent*; and it incurs minimal *delay* at both the source and end users. We implement our scheme in a smart phone and our experiment shows that the *computation cost* of the scheme is acceptable for low-power mobile devices. To realistically assess packet loss-resilience of the scheme, instead of assuming an independent packet loss model, we employ a Gilbert model [6] which closely characterizes packet loss behavior in wireless mobile networks. Our simulation results demonstrate that the scheme is able to achieve high authentication verification rates at much lower *communication overhead* compared to existing schemes.

2 Prerequisites

2.1 H.264/SVC Standard

The H.264/SVC standard encodes a video stream into 3-dimensional scalability using inter-layer prediction and hierarchical prediction structures. Inter-layer

[2] We focus on the Coarse Grain Scalability (CGS) layers.

prediction, which maximizes usage of lower layer information to improve the rate-distortion efficiency of a higher layer, provides spatial (resolution) and quality (PSNR, peak signal-to-noise ratio) scalabilities within an access unit (AU); it produces a base layer (BL) with the lowest resolution and PSNR, and multiple enhancement layers (EL) which improve the resolution and/or PSNR. Note that each spatial layer may have one or more quality layers and a higher spatial/quality layer utilizes a lower spatial/quality layer as reference, thus, we consider them as a single dimension and name it as "SQ scalability". An AU can be downscaled to a lower spatial/quality AU by discarding higher SQ layers.

The hierarchical prediction structure provides temporal scalability. It processes the AUs within a Group-of-Pictures (GOP) and group them into several temporal layers such that an AU in a higher temporal layer uses the preceding and succeeding AUs in the lower temporal layers as reference (see Fig. 5(a) for the hierarchical structure). Due to this structure, a GOP with T temporal layers consists of 2^{T-1} AUs. Note that temporal scalability is achieved by discarding all AUs in a higher temporal layer, which will reduce the frame rate by half. In Fig. 5(a), TL_1 is the temporal BL while $\mathsf{TL}_2, \mathsf{TL}_3, \mathsf{TL}_4$ are the temporal ELs.

Before an H.264/SVC bit stream is transported over the network, it is organized into Network Abstraction Layer units (NALU). NALUs are designed to form natural packet boundaries and can easily handle bit stream transcoding. NALUs are categorized into Video Coding Layer (VCL) NALUs and non-VCL NALUs. The VCL NALUs carry coded video data while non-VCL NALUs carry associated additional information. We make use of the Supplemental Enhancement Information (SEI) NALU [25], which is a non-VCL NALU that provides additional information to assist the decoding or bit stream manipulation process, to achieve format-compliance for our authentication scheme.

2.2 Erasure Correction Code (ECC)

Let k and n be two positive integers where $k < n$. A systematic (n, k) ECC consists of an encoder module and a decoder module. The encoder module takes a k-tuple of information symbols $X_k = x_1, x_2, \cdots, x_k$ and outputs an n-tuple of codeword $Y_n = y_1, y_2, \cdots, y_n = (x_1, x_2, \cdots, x_k, y_{k+1}, y_{k+2}, \cdots, y_n)$, where $y_{k+1}, \cdots, y_n$, are parity check symbols [13]. Given any k or more symbols in Y_n, the decoder module can output the original information tuple X_k [13]. In addition, all x_i and y_j have the same bit length, $1 \leq i \leq k$ and $1 \leq j \leq n$.

A double-ECC coding scheme (DECS), similar to that used in [19], is used in the proposed scheme. It uses an (n, k) and a $(2n-k, n)$ systematic ECC schemes, denoted $\mathsf{ECC}_{n,k}$ and $\mathsf{ECC}_{2n-k,n}$ respectively. The encoding and decoding functions of the DECS scheme are described below.

Encoding function $\mathsf{DECS\text{-}EN}_{n,k}(X_n)$ This function takes as input an n-tuple $X_n = (x_1, x_2, \cdots, x_n)$ and outputs $Z_n = (x_1 \| y_1, x_2 \| y_2, \cdots, x_n \| y_n)$, where "$\|$" denotes the concatenation of two symbols. The function works as follows:

1. Compute a $2n-k$-tuple $(x_1, x_2, \cdots, x_n, c_1, c_2, \cdots, c_{n-k}) \leftarrow \mathsf{ECC}_{2n-k,n}(X_n)$.
2. Divide $c_1 \| c_2 \| \cdots \| c_{n-k}$ into k symbols of equal length, i.e. $(d_1, d_2, \cdots, d_k)$.

3. Compute an n-tuple $(y_1, y_2, \cdots, y_n) \leftarrow \mathsf{ECC}_{n,k}(d_1, d_2, \cdots, d_k)$.
4. Output $W_n = (x_1 \| y_1, x_2 \| y_2, \cdots, x_n \| y_n)$.

Decoding function $\mathsf{DECS\text{-}DE}_{n,k}(Y_q)$ Suppose that a subset of W_n, $Y_q = (x_{i_1} \| y_{i_1}, x_{i_2} \| y_{i_2}, \cdots, x_{i_q} \| y_{i_q})$ is received, where $k \leq q \leq n$. The decoding function takes Y_q as input and outputs X_n with the following steps:

1. Use $\mathsf{ECC}_{n,k}$ to decode $(y_{i_1}, y_{i_2}, \cdots, y_{i_q})$ to obtain $(d_1, d_2, \cdots, d_k)$.
2. Divide $d_1 \| d_2 \| \cdots \| d_k$ into $(n-k)$ symbols of equal length, namely $(c_1, c_2, \cdots, c_{n-k})$.
3. Use $\mathsf{ECC}_{2n-k,n}$ to decode $(x_{i_1}, x_{i_2}, \cdots, x_{i_q})$ and $(c_1, c_2, \cdots, c_{n-k})$ to get $X_n = (x_1, x_2, \cdots, x_n)$ since $q + n - k \geq n$.

3 Proposed Scheme

We assume that a GOP G has T temporal layers where TL_1 is the temporal BL and TL_l ($l \in [2, T]$) is the l^{th} temporal EL, thus having a total of 2^{T-1} AUs, i.e., $G = [\mathsf{AU}_1, \mathsf{AU}_2, \cdots, \mathsf{AU}_{2^{T-1}}]$. For the SQ scalability, we assume each access unit has $M + 1$ SQ layers, then $\mathsf{AU}_i = \{\mathsf{Pre}_i, \mathsf{BL}_i, \mathsf{EL}_{i,1}, \mathsf{EL}_{i,2}, \cdots, \mathsf{EL}_{i,M}\}$, where Pre_i is a 4- or 5-byte non-VCL Prefix NALU carrying the spatial/quality/temporal ID of BL_i; BL_i is the SQ BL NALU and $\mathsf{EL}_{i,j}$ is the j^{th} SQ EL NALU of AU_i. The proposed scheme, presented below, exploits the advantages of [8] and [19] and is tailored explicitly for authenticating H.264/SVC bit streams.

3.1 Authentication of SVC Streams

An SVC bit stream is processed in GOPs, each with $n = 2^{T-1}$ AUs. For each AU_i, a hash chain is formed starting from the highest SQ EL NALU as in [8] and its final hash (hash of BL_i), denoted h_i, is regarded as the AU hash. Note that this hash chain approach makes the scheme proxy-transparent since SQ scalability is achieved by discarding SQ NALUs starting from the highest SQ EL NALU. Next, by concatenating hashes of all the AUs in a temporal layer we obtain the hash of the temporal layer, denoted H_l, for TL_l. A GOP hash is then computed over hashes of all temporal layers and a source signature is generated on the GOP hash. Then, all the hashes are encapsulated into SEI NALUs.

To protect against authentication data loss, we use a combination of DECS and packet replications. Unlike the scheme in [15] which uses packet interleaving to convert burst loss to random loss pattern, no interleaving is used in our scheme in order to reduce decoding delays. We also overcome a limitation in [15] where if the ECC is unable to recover AU hashes in temporal layer TL_l, AUs in $\mathsf{TL}_{l+1}, \cdots, \mathsf{TL}_T$, although received correctly and are usable in decoding, are unverifiable and must be discarded. In addition, as we will show later, the proposed scheme has a much lower communication overhead and does not adversely affect the PSNR of the underlying video stream. In the following, the source takes as input a GOP $G = [\mathsf{AU}_1, \mathsf{AU}_2, \cdots, \mathsf{AU}_{2^{T-1}}]$ and outputs an authenticated GOP supporting SQ and temporal scalability. Here, $\mathcal{H}(.)$ is a secure hash function.

Authentication of Spatial and Quality (SQ) Layers. To authenticate the SQ NALUs within an AU, the source takes as input an access unit $\mathsf{AU}_i = \{\mathsf{Pre}_i, \mathsf{BL}_i, \mathsf{EL}_{i,1}, \cdots, \mathsf{EL}_{i,M}\}$ and produces an authenticated access unit AU'_i.

Step 1. Let $h_{i,M} = \mathcal{H}(\mathsf{EL}_{i,M}); h_{i,j} = \mathcal{H}(\mathsf{EL}_{i,j} \| h_{i,j+1}),\ j \in [1, M-1]$.
Step 2. Output $\mathsf{AU}'_i = \mathsf{AU}_i \bigcup \{h_{i,j}\}_{j=1}^{M}$.

As mentioned earlier, the hash $h_{i,j}$ is put into an SEI NALU with the same (spatial, quality, temporal)-ID as $\mathsf{EL}_{i,j-1}$ for format compliance and transparent downscaling purposes. Note that the SQ dependency structure may sometimes be a directed acyclic graph depending on encoding configurations. In this case, we can employ a similar approach as in [15]. Thus, some NALUs may carry several hashes, but the total number of hashes within an AU remains the same.

Authentication of Temporal Layers and GOP. At this stage, the source has a group G' containing the set of access units $[\mathsf{AU}'_1, \mathsf{AU}'_2, \cdots, \mathsf{AU}'_{2^{T-1}}]$. In the following, the source further operates on the AUs within G' to support temporal scalability.

Step 3. For each AU'_i in G', compute the hash of AU'_i as $h_i = \mathcal{H}(\mathsf{BL}_i \| h_{i,1})$.
Step 4. Let n_l be the number of AUs in temporal layer TL_l, then $\mathsf{TL}_l = [\mathsf{AU}'_{i_1}, \mathsf{AU}'_{i_2}, \cdots, \mathsf{AU}'_{i_{n_l}}]$[3]. For each temporal layer TL_l ($l \in [1, T]$), compute the temporal layer hash as $H_l = \mathcal{H}(h_{i_1} \| h_{i_2} \| \cdots \| h_{i_{n_l}})$.
Step 5. Compute the GOP hash as $H_{GOP} = \mathcal{H}(H_1 \| H_2 \| \cdots \| H_T \| G_{id} \| S_{id})$, where G_{id} is the GOP identifier and S_{id} is the stream identifier. Compute $\sigma = Sign(H_{GOP})$ where $Sign(.)$ is a secure signature scheme.
Step 6. For each temporal layer TL_l ($l \in [1, T]$), generate the codeword:

$$(h_{i_1} \| y_{i_1}, h_{i_2} \| y_{i_2}, \cdots, h_{i_{n_l}} \| y_{i_{n_l}}) \leftarrow \text{DECS-EN}_{n_l,k_l}(h_{i_1}, h_{i_2}, \cdots, h_{i_{n_l}})$$

and let $\mathsf{AU}'_{i_b} \leftarrow \mathsf{AU}'_{i_b} \bigcup y_{i_b}$ ($1 \le b \le n_l$). Therefore, in TL_l, receiving any k_l SQ BL NALUs and the DECS code symbols y_{i_b} allows the user to recover the hashes of all AUs in TL_l and hence reconstruct H_l.
Step 7. Let $S = H_1 \| H_2 \| \cdots \| H_T \| \sigma$. Output the authenticated GOP as $G' = [\mathsf{AU}'_1, \mathsf{AU}'_2, \cdots, \mathsf{AU}'_{2^{T-1}}] \bigcup S$, where S is also placed in an SEI NALU.

In essence, the source generates a signature over all temporal layer hashes in a GOP, each temporal layer hash is computed over the hashes of all its AUs, and the hash of an AU carries the accumulated hashes of its SQ NALUs.

The source uses DECS in each temporal layer to protect the AU hashes. If a user wishes to obtain a particular temporal layer but some of its AUs are lost, then DECS can recover the lost AU hashes; otherwise, the user simply discards all the received AUs in that temporal layer and proceeds to verify AUs in all other temporal layers. Note that this cannot be achieved by the scheme in [15] because temporal layers are authenticated using a hash chain. In [15], the n_T AU hashes in TL_T are concatenated, segmented into k_{T-1} pieces, ECC encoded

[3] Note that $n_1 + n_2 + \cdots + n_T = 2^{T-1}$.

into n_{T-1} codewords and appended to the n_{T-1} AUs in TL_{T-1}. For each AU in TL_l, $l \in [1, T-1]$, the AU hash is then computed from the codeword it carries, along with its SQ NALU. The AU hash in TL_1 carries the accumulated hashes of higher layer AUs and it is regarded as the GOP hash. As a result, in any temporal layer TL_l, if less than k_l ECC codewords are received, the AUs hashes in TL_{l+1} cannot be recovered and consequently, all AUs in the higher temporal layers must be discarded. In addition, due to the hierarchical prediction structure, $n_{l-1} = \frac{1}{2}n_l$, ECC coding n_l hashes such that receiving k_{l-1} ($k_{l-1} < n_{l-1}$) can recover all n_l hashes resulted in a very high communication overhead. On the other hand, the use of DECS in our scheme within a temporal layer incurs a much lower communication overhead.

Because the authentication data S in Step 7 carries the signature, it is important for the SEI NALU carrying S be received, thus, we transmit several replications of this NALU to increase the probability for it to be received. In Sect. 4, we show that transmitting three copies of this NALU is enough for a close to 98% verification rate in typical wireless mobile networks.

3.2 Downscaling Process

Upon receiving an authenticated GOP $G' = [\mathsf{AU}'_1, \mathsf{AU}'_2, \cdots, \mathsf{AU}'_{2^{T-1}}]$ from the source, a proxy performs downscaling operation according to network bandwidth or user device capabilities by discarding some NALUs, starting from the highest SQ EL NALUs and/or the AUs in the highest temporal layer.

Suppose that the proxy desires to remove m ($m < M$) SQ EL NALUs and t ($t < T$) temporal layers, it first discards AUs in temporal layers $\mathsf{TL}_T, \mathsf{TL}_{T-1}$, $\cdots$, TL_{T-t+1}. Let $T' = T - t - 1$, then $G' = [\mathsf{AU}'_1, \mathsf{AU}'_2, \cdots, \mathsf{AU}'_{2^{T'}}]$. For each AU'_i ($i \in [1, 2^{T'}]$), the proxy discards $\mathsf{EL}'_{i,M}, \mathsf{EL}'_{i,M-1}, \cdots, \mathsf{EL}'_{i,M-m+1}$ to form the new $\mathsf{AU}''_i = \{y_i, \mathsf{Pre}_i, \mathsf{BL}_i, h_{i,1}, \mathsf{EL}_{i,1}, h_{i,2}, \mathsf{EL}_{i,2}, \cdots, h_{i,M'}, \mathsf{EL}_{i,M'}, h_{i,M'+1}\}$, where $M' = M - m$. Finally, the proxy also discards the corresponding SEI NALUs and transmits the downscaled GOP $G'' = [\mathsf{AU}''_1, \mathsf{AU}''_2, \cdots, \mathsf{AU}''_{2^{T'}}]$ to end users. Note that in this process, the proxy simply discards NALUs and/or AUs without needing to understand the authentication mechanism.

3.3 Verification Process

A user performs the following verification steps as she receives a GOP $\tilde{G} = [\tilde{\mathsf{AU}}_1, \tilde{\mathsf{AU}}_2, \cdots, \tilde{\mathsf{AU}}_{2^{T'}}]$.

Step 1. If the SEI NALU containing $\tilde{S}$ is received, go to Step 2. Otherwise, $\tilde{G}$ is not verifiable and is discarded.

Step 2. Parse $\tilde{\mathsf{AU}}_i$ into $\{y_i, \mathsf{Pre}_i, \mathsf{BL}_i, h_{i,1}, \mathsf{EL}_{i,1}, \cdots, h_{i,M'}, \mathsf{EL}_{i,M'}\}$ if $M' = M$, or into $\{y_i, \mathsf{Pre}_i, \mathsf{BL}_i, h_{i,1}, \mathsf{EL}_{i,1}, \cdots, h_{i,M'}, \mathsf{EL}_{i,M'}, h_{i,M'+1}\}$ if $M' < M$.

Step 3. For each $\tilde{\mathsf{AU}}_i$, compute its hash starting from the M'^{th} SQ EL NALU, i.e. $\tilde{h}_{i,M'} = \mathcal{H}(\mathsf{EL}_{i,M'})$ if $M' = M$, or $\tilde{h}_{i,M'} = \mathcal{H}(\mathsf{EL}_{i,M'} \| h_{i,M'+1})$ if $M' < M$, and in other cases, $\tilde{h}_{i,j} = \mathcal{H}(\mathsf{EL}_{i,j} \| \tilde{h}_{i,j+1})$; finally, $\tilde{h}_i = \mathcal{H}(\mathsf{BL}_i \| \tilde{h}_{i,1})$.

Step 4. Group the AUs based on the temporal layer ID, compute the temporal layer hash as $\tilde{H}_l = \mathcal{H}(\tilde{h}_{i_1} \| \tilde{h}_{i_2} \| \cdots \| \tilde{h}_{i_{n_l}})$. In the case of packet loss, as long as k_l or more SQ BL NALUs and SEI NALUs are received, the user performs DECS-DE to recover the missing AU hashes. If less than k_l SQ BL NALUs and SEI NALUs are received, discard the remaining received NALUs.

Step 5. Compute the GOP hash as $\tilde{H}_{GOP} = \mathcal{H}(\tilde{H}_1 \| \tilde{H}_2 \| \cdots \| \tilde{H}_T \| G_{id} \| S_{id})$ where $\tilde{H}_{T'+1}, \cdots, \tilde{H}_T$ are recovered from $\tilde{S}$.

Step 6. Verify the signature σ against $\tilde{H}_{GOP}$. If the verification outputs true, accept $\tilde{G}$; otherwise, reject it.

4 Performance Evaluation

We implement and integrate the proposed scheme into an open source SVC decoder (Opensvc [3]) on the Android V1.5 platform. We utilize the open source ECC [21] and the signature software [18] which, respectively, implement the Reed-Solomon Code [22] and the RSA signature [23] with the hash function SHA-1 [17]. We cross compile these C source codes for the ARM-based smart phone on a X86-based desktop. The binary code is encapsulated in JNI (JAVA Native Interface) and JAVA renders the User Interface.

In our implementation, we slightly modify the proposed scheme as follow. Firstly, we note that 0x00000001 and 0x000001 are forbidden sequences in the H.264/SVC bitstream because they mark the NALU boundary [10]. If such sequences occur in authentication data, we replace them as shown in Table 1 and recover them as the NALU is read from the stream. Secondly, knowledge of the position of each AU in a GOP is necessary for DECS to recover the AU hashes in the event of packet loss. We insert this position information into the SEI NALU corresponding to the SQ BL NALU.

Table 1. Forbidden sequences and their replacements

Forbidden Sequence	0X000000	0X000001	0X000002	0X000003
Replacement	0X00000300	0X00000301	0X00000302	0X00000303

4.1 Loss-Resiliency and Communication Overhead

A video stream can tolerate loss at the cost of a lower PSNR, but an authenticated stream has a smaller window of tolerance because all data in a GOP must be discarded if it cannot be verified. In the following, we study loss-resiliency of the proposed scheme by measuring its *verification rate*, defined as the ratio of "the number of verifiable NALUs" to "the number of received NALUs", under the Gilbert model [6] (see Fig. 1). In contrast to the independent loss model, the Gilbert model is a more realistic approximation to real wireless mobile networks. In this model, the two states have different packet loss probabilities. The probabilities P_{gg}, P_{gb}, P_{bg}, P_{bb} respectively represents the state transition probabilities of "Good-to-Good", "Good-to-Bad", "Bad-to-Good" and "Bad-to-Bad".

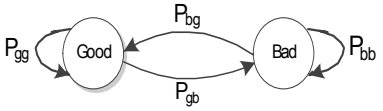


Fig. 1. Gilbert channel model

In [2], the authors formulated a packet loss model aimed at H.264 video transmission over the IEEE 802.11g Wireless LANs. This model assumes no packet loss in "Good" state and some packet loss in "Bad" state. In our experiment, we assume that packet is always lost in the "Bad" state and adapt their measurement data as our Gilbert model parameters (listed in Table 2). The four scenarios in Table 2 respectively depicts a wireless transmission for low (500 kbps) and high (2000 kbps) bitrate video streams under a relatively small (300 Bytes) and large (1200 Bytes) Maximum Transmission Unit (MTU) size.

Table 2. Gilbert model parameters

	bitrate(kbps)	MTU(bytes)	P_{gb}	P_{bg}	P_{loss}	β	% loss
Scenario 1	500	300	7.43e-04	4.92e-03	1.31e-01	204	97%
Scenario 2	500	1200	2.90e-03	2.89e-02	0.91e-01	34	65%
Scenario 3	2000	300	2.50e-04	2.49e-03	0.91e-01	402	48%
Scenario 4	2000	1200	7.30e-04	6.79e-03	0.97e-01	148	71%

The average packet loss probability P_{loss} increases with burst loss length β (in packets), packet rate and MTU size. From Table 2, Scenario 1 has the highest P_{loss} while Scenarios 2, 3 and 4 exhibit similar P_{loss}. This is because Scenario 1 has a larger β and a higher packet rate compared to Scenarios 2 and 4, while between the latter two, Scenario 4 has a larger β, packet rate as well as MTU size. The lower P_{loss} in Scenario 3 could be attributed to its small MTU size, however, its high packet rate results in the largest β. A large β in a video stream causes a lower verification rate for low bitrate streams compared to high bitrate streams because of a larger percentage of video/authentication data that is lost (% loss) in one burst. Thus, for low bitrate streams, the verification rate will be lower in Scenario 1 compared to Scenario 2 due to a larger % loss in Scenario 1. For high bitrate streams, the verification rate will be lower in Scenario 4 compared to Scenario 3 even though Scenario 3 has the largest β. This is due to the smaller MTU size in Scenario 3 where even a larger β does not adversely affect the % loss as that in Scenario 4. The above observation is in agreement with the results of our computer simulations.

We simulate the verification rates for the proposed scheme in all scenarios. In our simulation, we assume the following:

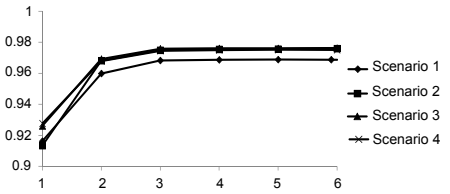


Fig. 2. Probability of receiving NALU S vs. number of copies of NALU S

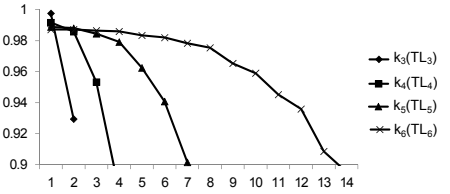


Fig. 3. Probability of temporal layer authentication vs. its DECS parameter

1. The channel does not have a retransmission mechanism.
2. Large NALUs are divided into small packets according to the maximum transmission unit (MTU).
3. The entire NALU will be discarded when one or more of its packets are lost.
4. Unless otherwise stated, the GOP size $|\mathsf{GOP}| = 32$. Also, we assume the scheme in [15] computes one signature over 5 GOPs.
5. Since the SEI NALU containing S carries the temporal layer hashes and the GOP signature, we transmit it multiple times to increase the probability of it being received. As shown in Fig. 2, repeating this NALU three times guarantees at least a 0.98 probability of it being received in all four scenarios.
6. Since DECS is used in each temporal layer to increase the probability of a temporal layer being authenticated, we choose the DECS parameter k_l for each temporal layer TL_l that maximizes this probability. Figure 3 shows the probabilities of authenticating temporal layers in Scenario 1, which has the worst performance among the four scenarios. We see that setting $(k_1, k_2, k_3, k_4, k_5, k_6) = (1, 1, 1, 2, 5, 10)$ gives us a ≈ 0.98 probability of authentication, where $k_1 = k_2 = 1$ is due to the hierarchical prediction structure in H.264/SVC.

In the following, we examine the loss-resiliency, communication overhead, computation cost and delay of the proposed scheme assuming the parameters calibrated above.

Loss Resiliency w.r.t. Number of SQ Layers. To verify the authenticity of an AU, the AU hash must be reconstructed from hash-chaining the AU's SQ NALUs. However, in the event of loss, the hash chain can be broken. An intuitive solution is to protect the SQ NALU hashes using DECS. More specifically, for each $\mathsf{EL}_{i,j}$ (i.e. the j^{th} SQ EL NALU of AU_i), use DECS to protect $h_{i,j+1} \| h_{i,j}$ such that if $\mathsf{EL}_{i,j}$ is lost, $h_{i,j+1}$ can be used to authenticate $\mathsf{EL}_{i,j+1}$ and $h_{i,j}$ can be used to reconstruct the chain. Assuming DECS computation is negligible, this method incurs a higher communication overhead, i.e. instead of having one hash per SQ NALU, there are effectively three hashes per SQ NALU.

Nevertheless, we simulate the verification rate of the proposed scheme *with* and *without* SQ NALU hash protection for Scenario 1 with the above measure. Figure 4 shows that the verification rate drops when the number of spatial layers increases. Interestingly, the verification rate is not adversely affected whether or not the SQ NALU hashes are protected. Thus, for the interest of communication

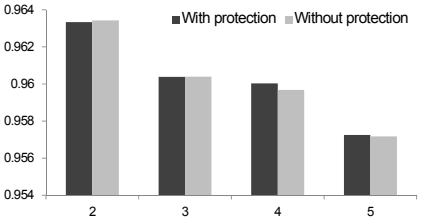
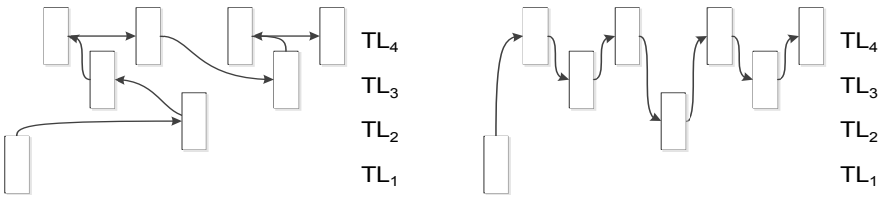


Fig. 4. Verification rate with and without SQ NALU hash protection vs. number of SQ layers

overhead, we do not protect SQ NALU hashes. However, we note that in [15], every SQ NALU hash is replicated twice to counter packet loss.

Loss Resiliency w.r.t. Transmission Order. We further study the verification rate of the proposed scheme by considering the bit stream transmission order. For an SVC bit stream generated by JSVM (Joint Scalable Video Model) [9], there are two types of stream orders: 1) video playback (Tx_Play) and 2) preorder traversal (Tx_PT) (see Fig. 5(a) and 5(b)) based on source's encoding delay configuration D_{enc}. When D_{enc} is small, AUs will be encoded and transmitted in Tx_Play order; when D_{enc} is large enough, Tx_PT order will be used. Normally, Tx_PT is used because it has less intra-frame redundancy.

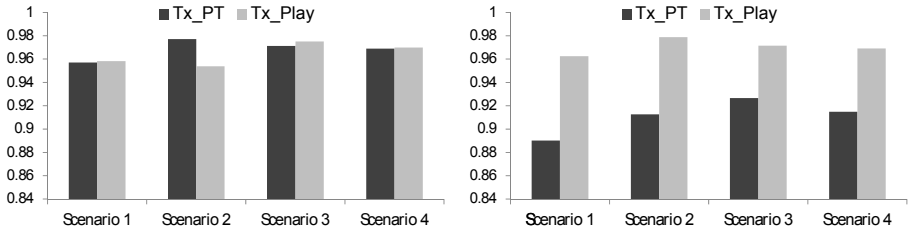


(a) Preorder traversal order, Tx_PT (b) Video playback order, Tx_Play

Fig. 5. Types of stream orders

For the sake of comparison, we study the verification rate of our scheme and that of the scheme in [15] in both transmission orders. Note that unlike [15], the proposed scheme does not support MGS scalability. Therefore, in our implementation of the scheme in [15] (and also [16]), we do not consider MGS scalability. Figures 6(a) and 6(b) show that with five SQ layers, both schemes have similar verification rates for Tx_Play while the proposed scheme has a higher verification rate for Tx_PT. This is because as mentioned earlier, the scheme in [15] authenticates temporal layers using a hash chain where the final hash is signed. Note that in [15], the "temporal base layer" is a combination of TL_2 and TL_1. In Tx_PT, the AUs in TL_2 and TL_1 are transmitted consecutively, which means that they are likely to be lost in one burst. Moreover, since temporal

layers are authenticated using a hash chain, the loss of the lower layers renders the higher layers unauthenticated.



(a) Proposed scheme (b) The scheme in [15]

Fig. 6. Verification rate in different transmission orders

Communication Overhead. The proposed scheme achieves a high loss-resiliency with a sufficiently low communication overhead. Considering a GOP size of 32 and that each AU has five SQ layers (including SQ base layer), the proposed scheme appends a total of 3808 Bytes per GOP while the scheme in [15] appends a total of 11620 Bytes per GOP; for a stream with a specific bitrate, the amount of authentication data results in a lower effective video bitrate, thus a lower PSNR (see Fig. 7).

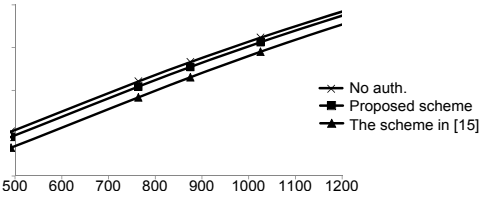


Fig. 7. PSNR for the proposed scheme and the scheme in [15] vs. bitrate

4.2 Computation Cost and Delay

The primary computation cost is incurred by signature verification and hash computation. The RSA signature verification time on a Samsung S5830@800MHz smart phone is 5.65ms per verification while SHA-1 computation time is proportional to the stream bitrate since hash function processes a block of 64-bit input at a time. Therefore, for $|\mathsf{GOP}| = 32$, the proposed scheme costs about 5.92ms to 6.72ms per GOP on the smart phone when the bitrate is varied from 500 kbps to 2000 kbps. When the stream "Foreman" is encoded with one quality enhancement layer at resolution 352×288 on this platform, the decoding time is about 85ms per frame. The proposed scheme only costs $\frac{6.72}{85 \times 32} = 0.2\%$ of the decoding time. Though the scheme in [15] incurs even smaller computation cost because there is only one signature verification for five GOPs, the computation

costs in both schemes are negligible in a typical smart device today. Table 3 compares the computation cost, source and user delay of the proposed scheme with those of the scheme in [15].

Table 3. Delay and computation cost comparison

	Source Delay	User Delay	User Computation Cost
Proposed scheme	1 GOP	1 GOP	0.2%
Scheme in [15]	5 GOPs	$1-5$GOPs	0.08%

5 Conclusion

We proposed a scheme for authenticating H.264/SVC video streams over packet-lossy networks. The scheme preserves scalability of H.264/SVC streams in the spatial, quality and temporal dimensions and have several highly desirable features such as proxy-transparency, low communication overhead and low delay. Packet loss resiliency of the scheme was studied by simulating transmission of authenticated streams over a realistic bursty wireless mobile network characterized using a Gilbert model. Our experimental results showed that the proposed scheme achieves high verification rate over typical non-stationary, bursty packet-lossy wireless mobile networks. The proposed scheme was implemented and integrated with an SVC decoder on an Android platform, and our measurement indicated that the computation cost due to authentication is negligible.

Acknowledgement. This research is supported by A*STAR SERC Grant No. 102 101 0027 in Singapore.

References

1. Deng, R.H., Yang, Y.: A study of content authentication in proxy-enabled multimedia delivery systems: Model, techniques and applications. ACM Transactions on Multimedia Computing, Communications and Applications 5(4), 1–20 (2009)
2. Ferre, P., et al.: Packet loss modelling for H.264 video transmission over IEEE 802.11g wireless LANs. In: IEEE 13th International Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS (2005)
3. Geeknet Inc.: Open SVC decoder, `http://sourceforge.net/apps/mediawiki/opensvcdecoder/index.php?title=Main_Page`
4. Gennaro, R., Rohatgi, P.: How to Sign Digital Streams. In: Kaliski, B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 180–197. Springer, Heidelberg (1997), `http://dx.doi.org/10.1007/BFb0052235`
5. Gentry, C., et al.: End-to-end security in the presence of intelligent data adapting proxies: The case of authenticating transcoded streaming media. IEEE Journal on Selected Areas in Communications 23(2), 464–473 (2005)
6. Gilbert, E.: Capacity of a burst-noise channel. Bell System Technical Journal 39, 1253–1265 (1960)
7. Golle, P., Modadugu, N.: Authenticating streamed data in the presence of random packet loss. In: Proceedings on Network and Distributed Systems Security Symposium (NDSS 2001), pp. 13–22 (2001)

8. Hefeeda, M., Mokhtarian, K.: Authentication schemes for multimedia streams: Quantitative analysis and comparison. ACM Transactions on Multimedia Computing, Communications and Applications 6(1), 1–24 (2010)
9. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG: Joint scalable video model, JVT-X202 (2007)
10. ITU-T Rec. H.264: H.264 advanced video coding for generic audiovisual services (2003)
11. Kaced, A.R., Moissinac, J.C.: Multimedia content authentication for proxy-side adaptation. In: Proceedings of International Conference on Digital Telecommunication (ICDT), pp. 62–66 (2006)
12. Li, T., Wu, Y.: Adaptive stream authentication for wireless multimedia communications. In: Proceedings of IEEE Wireless Communications and Networking Conference (WCNC), pp. 2613–2618 (2007)
13. Lin, S., Costello, D.J.: Error Control Coding: Fundamentals and Applications, 2nd edn. Prentice-Hall, NJ (2004)
14. Miner, S., Staddon, J.: Graph-based authentication of digital streams. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 232–246 (2001)
15. Mokhtarian, K., Hefeeda, M.: End-to-end secure delivery of scalable video streams. In: Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2009), pp. 79–84 (2009)
16. Network Systems Lab: svcAuth - NSL, `http://nsl.cs.sfu.ca/wiki/index.php/svcAuth`
17. NIST: Federal Information Processing Standard (FIPS) 180-2, Secure Hash Standard (2002)
18. Offspark.com: Polar SSL, `http://polarssl.org/`
19. Pannetrat, A., Molva, R.: Efficient multicast packet authentication. In: Proceedings on Network and Distributed Systems Security Symposium, NDSS 2003 (2003)
20. Park, J.M., Chong, E.K.P., Siegel, H.J.: Efficient multicast stream authentication using erasure codes. ACM Transactions on Information and System Security 6(2), 258–285 (2003)
21. Plank, J.S., Simmerman, S., Schuman, C.D.: Jerasure: A library in C/C++ facilitating erasure coding for storage applications - Version 1.2. Tech. Rep. CS-08-627, University of Tennessee (August 2008)
22. Reed, I.S., Solomon, G.: Polynomial codes over certain finite fields. Journal of the Society for Industrial and Applied Mathematics (SIAM) 8(2), 300–304 (1960)
23. Rivest, R.L., Shamir, A., Adleman, L.: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
24. Song, D., Zuckerman, D., Tygar, J.D.: Expander graphs for digital stream aauthentication and robust overlay networks. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 258–270 (2002)
25. Wenger, S., Wang, Y.K., Schierl, T., Eleftheriadis, A.: RTP payload format for scalable video coding (2011)
26. Wong, C., Lam, S.: Digital signatures for flows and multicasts. IEEE Transactions on Networking 7(4), 502–513 (1999)
27. Wu, Y., Deng, R.H.: Scalable authentication of MPEG-4 streams. IEEE Transactions on Multimedia 8(1), 152–161 (2006)
28. Zhang, Z., Sun, Q., Wong, W.C.: A proposal of butterfly-graph based stream authentication over lossy networks. In: Proceedings of IEEE International Conference on Multimedia and Expo (ICME 2005), 4 pages (2005)

The Performance of Public Key-Based Authentication Protocols

Kaiqi Xiong

College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, NY 14623 USA

Abstract. Kerberos has revolved over the past 20 years. Kerberos and its variants have been extensively used in a variety of commuting systems since 1999. Among them, there have been several techniques and protocols to integrate public key cryptography into Kerberos. Public-Key Cross Realm Authentication in Kerberos (PKCROSS) is one of these protocols. It has been proposed to simplify the administrative burden of maintaining cross-realm keys so that it improves the scalability of Kerberos in large multi-realm networks. Public Key Utilizing Tickets for Application Servers (PKTAPP) is another protocol that has been suggested to improve the scalability issue of PKCROSS. Performance evaluation is a fundamental consideration in the design of security protocols. But, the performance of these two protocols has been poorly understood in a large-scale network. In this paper, we present an efficient way to study the performance of PKCROSS and PKTAPP. Our thorough performance analysis of these two protocols shows that PKTAPP does not scale better than PKCROSS. In this paper, we report our recent results of when PKCROSS still outperforms than PKTAPP in multiple remote realms.

Keywords: Public-key infrastructure (PKI), Kerberos, Authentication, Performance, Transaction time.

1 Introduction

In the last few years we have witnessed an explosive growth in the usage of Internet and cloud collaborative applications such as video and audio conferencing, replicated servers and databases, and in particular cloud service composition in that services components from several universe service providers can be flexibly integrated into a composite service regardless of their location, platform, and execution speed [6]. To ensure quality of services, these service providers are required to work together. The rapid growth in collaborative applications has heightened the need for a reliable group communication system. The system, in turn, is impossible to be reliable without group authentication.

Kerberos [24] is a mature, reliable, secure network authentication protocol that allows a client to prove its identity to a server without sending confidential data across the network. Partitioning of the world into realms served by different application servers is a way to improve the scalability of Kerberos. In each realm, Kerberos consists of a

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 206–219, 2012.

client, application servers and a key distribution center (KDC). The client may represent a group of business users who request services from application servers. The KDC maintains a shared symmetric key with every client and the application servers in the realm. In case KDC is compromised (i.e., a single failure), all the symmetric keys will be divulged to the attacker and will have to be revoked. Recovering from such a compromise requires the re-establishment of new shared keys with the client and the application servers in the realm. Such a recovery is very costly in terms of time, effort and financial resources.

Public key cryptography has been extended to support Kerberos since it simplifies the distribution of keys in Kerberos. It eliminates a single point of failure. Integrating public key cryptography into Kerberos represents the enhancements of the current Kerberos standard. Several Kerberos-based authentication techniques using public-key cryptography have been proposed in the last decade. Among them include Public-Key Cross Realm Authentication in Kerberos (PKCROSS) [22] and Public-key Cryptography for Initial Authentication in Kerberos (PKINIT) [34]. Moreover, the scalability of network security infrastructures is becoming a serious concern as the explosive growth of collaborative applications such as Web services continues unabated. Public key based Kerberos for Distribution Authentication (PKDA) [32] and Public Key Utilizing Tickets for Application Servers (PKTAPP, a.k.a. KX.509/KCA) [25] and [26] have been proposed to enhance the security and scalability of Kerberos.

PKINIT is a core specification among these Internet drafts. Both PKCROSS and PKTAPP use variations of PKINIT message types and data structures for integrating public key cryptography with Kerberos in different authentication stages. PKTAPP was originally introduced as PKDA. It implemented PKDA using the message formats and exchanges of PKINIT. Microsoft has adopted the Internet draft specification of PKINIT for the support of public key cryptography in the Windows 2000 and 2003 implementations of Kerberos [15]. It has its own protocol that is the equivalent of KX509/KCA. There were preliminary discussions regarding an adoption of a common protocol between the Kerberos WG and Microsoft. The MIT Kerberos consortium will drive these discussions [5]. According to Altman [4], the standardization of PKCROSS and PKTAPP will be the next for Kerberos and PKI integration. The Kerberos Consortium at MIT was formed in September 2007 led by initial board members from Apple, Google, MIT, Microsoft and Sun and sponsored by 19 leading companies, universities and government agencies [8]. It is expected that Kerberos-based authentication and authorization will be as ubiquitous as TCP/IP-based networking itself. PKINIT and PKCROSS are listed as Projects 10 and 11 in the Kerberos Consortium's proposal respectively [27]. As stated above, PKCROSS and PKTAPP use variations of PKINIT message types and data structures in the design of the authentication protocols. We believe that PKCROSS and PKTAPP will revive soon. Hence, this paper only considers these two notable techniques: PKCROSS and PKTAPP. The performance analysis of these two protocols also provides the guidance for understanding their variants.

It has been argued that PKCROSS would not scale well in large networks in Sirbu and Chuang [32]. PKTAPP was proposed to improve the scalability of PKCROSS. However, the actual costs associated with these techniques have been *poorly understood* so far. Although PKTAPP is shown to poorly perform when there are two or more application

servers in one remote realm in Harbitter and Menasce [21], it remains *unknown* which technique performs better in a large network where application servers are within *multiple* KDC remote realms that are typical in many applications.

Performance evaluation is a fundamental consideration in the design of protocols. Two conventional approaches have been proposed to analyze the performance of a security protocol. The first one is to implement under study protocols and then take their measurements in real systems (for example, see Amir et al. [1] and [31]). This approach seems very good but it is time-consuming. Whenever the under-study protocol is changed, the updated protocol has to be re-implemented and the measurement data of the protocol implementation has been re-collected. This approach cannot be applied to our study of PKCROSS and PKTAPP. The implementation of original PKCROSS and PKTAPP is not available due to lack of resource and funding [5]. (Note: the implementation of PKTAPP's variant KX.509) was just released last year [25].) The second one is to count the number of operations and then compute their corresponding costs within a under study protocol (for example, see Amir et al. [2], [3], and Steiner et al. [33]). This approach is straightforward. It has widely used by researchers but it is not very easy to be used when the protocol becomes complicated such as the case of multiple KDC remote realms. Furthermore, in this case, an authentication request cannot often be processed fast enough, so it should wait in a queue. However, the second approach does not consider the waiting time of an authentication request in a queue. Hence, in order to efficiently and effectively analyze the performance of a protocol, we use a queueing network model as our protocol evaluation tool.

This paper first presents a thorough performance evaluation of PKCROSS and PKTAPP in terms of computational and communication costs. Although the complexity of message exchanges in multiple KDC remote realms, we figure out the number of secret, private and public key operations required in PKCROSS and PKTAPP. Then, we demonstrate their performance difference using open queueing networks in which we gain a better understanding of these two techniques compared to the approach of only counting the number of operations in the two techniques. An in-depth analysis of these two techniques shows that PKTAPP does not scale better than PKCROSS. This is a contrary result as expected by PKTAPP protocol designers. It should be pointed out that our proposed performance methodology is an effective way to analyze these authentication techniques which can be thus extended to analyze other security protocols as well.

The rest of this paper is organized as follows. In Section 2, we begin by giving an in-depth discussion of PKCROSS and PKTAPP and then presents their performance evaluation using queueing theory. The related work is discussed in Section 3. We finally conclude our discussion in Section 4.

2 PKCROSS and PKTAPP

Kerberos [24] is a network authentication protocol for providing a secure communication between a user workstation (or called a client) and application servers that was developed at the Massachusetts Institute of Technology in 1988. The latest version of Kerberos is Version 5. It divides the world into realms, each with user workstations, a

single primary KDC, back-up KDCs, and application servers in which the KDC is a trusted intermediary.

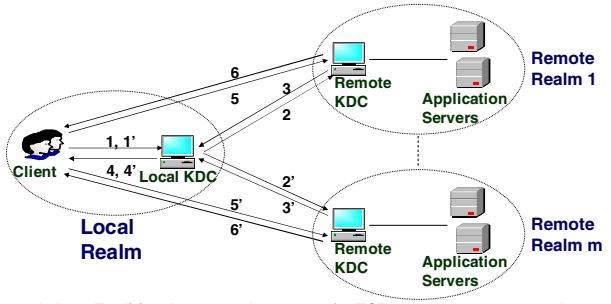
In the Kerberos protocol, the client engages in a multiple-step authentication to obtain access to the application server whereby the client first obtains a relatively short lived credential: a Ticket-Granting Ticket (TGT) from the Authentication Service running on a (KDC), and then obtain a session ticket for a particular application server by presenting the TGT to a centralized Ticket-Granting Service (TGS) running on the KDC. The client presents the session ticket to the application server for authenticating herself/himself by showing knowledge of a secret session key. The secret session key was securely passed to the client by the KDC. Kerberos is stateless. It is extremely valuable from the scalability point of view. Cross-realm authentication is necessary when a client and an application server with different network domains fall into different Kerberos realms.

It is well-known that a public key security system is easier to administer, more secure, less trustful, and more scalable than a symmetric key security system. Public key security doesn't use a trusted key management infrastructure since the burden of key management falls to public key clients [12]. In a public key infrastructure, public key clients need constantly and vigorously check the validity of the public keys that they use. Public key cryptography shifts the burden on key management from the KDC/TGS in Kerberos to its Certificate Authority (CA) who may be considered as a trusted intermediary. CA issues a public key certificate that is relatively long lived credential. The burden is determined by the number of times that clients want to authenticate to application servers in Kerberos. It might not be affordable in time-sensitive applications if one large-scale PKI deployment is needed for group authentication in a large network.

2.1 Protocol Analysis

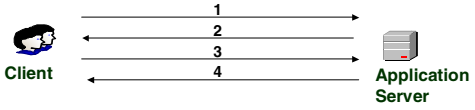
PKCROSS [22] is one notable protocol of integrating public key cryptography with Kerberos to address the problem of network authentication among the client and the application servers in a large number of realms. Figure 1 illustrates the authentication steps of PKCROSS. The cross realm KDC to KDC authentication is achieved by using public key cryptography.

First, just like Kerberos the KDC in PKCROSS is burdened by the need to constantly renew short-lived TGTs, and TGS must be involved whenever a client wants to request a service from an application server. Thus, if a large number of users in a single realm request services that may be a typical case in Web services, the KDC in the realm becomes a serious single point of failure due to a KDC compromise and possibly a performance bottleneck. Recovering such a compromise requires the re-establishment of secret keys for all users within this realm. When a number of users is large, such a recovery is very costly. PKINIT facilitates public-key based authentication between users and their local KDC. Hence, one possible solution to eliminating the single point of failure is to combine PKCROSS with PKINIT, and thus public-key based authentication is integrated with the *entire* Kerberos environment. However, it is easy to see that this solution is not feasible for time-sensitive applications, since the users might suffer from a long delay or even denial of services due to a high computational cost for the calculation of a public key used in the entire environment.



1 and 1'. Traditional cross-realm request for TGT to access a remote realm
2 and 2'. AS_REQ with PKCROSS flag set
3 and 3'. AS_REP with PKCROSS ticket encrypted using local KDC's public key
4 and 4'. Traditional reply with cross-realm TGT using SKC
5 and 5'. Ticket-Granting Service Request (TGS-REQ)
6 and 6'. Ticket-Granting Service Reply (TGS-REP)

Fig. 1. The PKCROSS Message Flow



1. A standard Kerberos version 5 message that includes PA-PK-AS-REQ pre-authentication field
2. A standard Kerberos version 5 message that includes PA-PK-AS-REP pre-authentication field
3. AS_REQ in which the client requests a service from the application server, using the traditional secret-key procedure defined in RFC1510
4. AS_REQ in which the client receives a reply from the application server, using the traditional secret-key procedure defined in RFC1510

Fig. 2. The PKTAPP Message Flow

Second, in PKCROSS the local KDC of the client issues all short-lived TGTs and all session tickets in its realm, and communicates with a remote KDC. Hence, it can easily become a performance bottleneck since all these authentication transactions have to transit the KDC. Thus, PKTAPP [26] has been proposed to address the issue. Figure 2 shows the message exchange of PKTAPP. The PKTAPP technique allows the client to communicate directly with the application servers so that the number of messages between them is reduced in an authentication process. But, as is seen in [21], even though PKTAPP requires fewer message exchanges than PKCROSS during client authentication with remote application servers, PKCROSS outperforms PKTAPP when two or more application servers are in a single remote realm. This is because PKTAPP requires more public-key message exchanges than PKCROSS that only requires one pair of public-key message exchanges. Below, we are going to study their performance in multiple remote realms.

We employ the first approach to analyzing the performance of the two techniques in which we are required to carefully compute the number of secret, private and public operations used in PKCROSS and PKTAPP in the case of a multiple remote realm.

Table 1. The Operations of Encryption and Decryption When n Application Servers are in m Remote Realms

Protocols	Entities	# Secret Key	# Private Key	# Public Key
PKTAPP	Client	$2n$+1	n+1	$3n$
	Application server	$3n$+1	n	$4n$
	Total	$5n$+2	$2n$+1	$7n$
PKCROSS	Client	m+6	0	0
	Local KDC	5	m+1	$3m$
	Remote KDC	$3m$+n	m	$4m$
	Application server	$3n$	0	0
	Total	4(n+m)+11	$2m$+1	$7m$

Table 1 summarizes the encryption and decryption operations performed in these two techniques where m and n are the number of remote realms and the number of applications servers within these realms respectively. Denote by c_j the computational times per secret, private, or public key operation respectively where $j = 1, 2, 3$. Then, $c_1 < c_2 < c_3$. Let $f_j(n, m)$, $j = 1, 2$, be the total computational times of encryption and decryption operations in PKTAPP and PKCROSS. Thus, from Table 1 we have that

$$f_1(n, m) = (5c_1 + 2c_2 + 7c_3)n + 2c_1 + c_2$$
$$f_2(n, m) = 4c_1 n + (4c_1 + 3c_2 + 7c_3)m + 11c_1 + c_2$$

Note that $f_1(n, m)$ does not depend on m, which can be hence abbreviated as $f_1(n)$. Then, we have the following proposition.

Proposition 1. *For each authentication, PKCROSS requires less computational time than PKTAPP if and only if the number of application servers n is more than $\lceil m + \frac{3(m+3)c_1}{c_1+2c_2+7c_3} \rceil$, or the number of remote realms m is less than $\lfloor n - \frac{3(n+3)c_1}{4c_1+2c_2+7c_3} \rfloor$.*

Proof. According to $f_1(n, m)$ and $f_2(n, m)$, their difference is given by $f_1(n, m) - f_2(n, m) = -9c_1 + (c_1 + 2c_2 + 7c_3)n - (4c_1 + 2c_2 + 7c_3)m$. Hence, $f_1(n, m) - f_2(n, m) \geq 0$ if and only if $-9c_1 + (c_1 + 2c_2 + 7c_3)n - (4c_1 + 2c_2 + 7c_3)m \geq 0$, which implies Proposition 1.

It is anticipated that the client is connected to the local KDC by a LAN, the client and the local KDC are connected to a remote KDC by a WAN, and a remote KDC and its application servers are connected by a WAN. Assume that all WANs have an identical communication time and a local area network (LAN) has a neglected communication time compared to a WAN. From Figures 1 and 2, we note that the number of WAN communications are $4n$ for PKTAPP and $4m + 2n$ for PKCROSS. Let $g_j(n, m)$, $j = 1, 2$, be the transaction times of PKTAPP and PKCROSS. *The transaction time* is defined as the computational time of the total encryption and decryption operations plus the communication time per authentication in a technique. Also, denote by d the time spent in a WAN communication. Then, the following conclusion holds.

Proposition 2. *For each authentication, PKCROSS uses less transaction time than PKTAPP if and only if the number of application servers n is more than $m + \lceil \frac{(3c_1+2d)m+9c_1}{c_1+2c_2+7c_3+2d} \rceil$.*

Proof. For $j = 1, 2, g_j(n, m)$ can be computed by $g_1(n, m) = f_1(n)+4n$ and $g_2(n, m) = f_2(n, m) + 4m + 2n$ and thus their difference is given by $g_1(n, m) - g_2(n, m) = (c_1 + 2c_2 + 7c_3 + 2d)n - (4c_1 + 2c_2 + 7c_3 + 4d)m - 9c_1$ which easily derives Proposition 2.

Note that if $n = 1$ (so, $m = 1$), $m + \lceil \frac{(3c_1+2d)m+9c_1}{c_1+2c_2+7c_3+2d} \rceil > 1 = n$, and if $m = 1$, $m + \lceil \frac{(3c_1+2d)m+9c_1}{c_1+2c_2+7c_3+2d} \rceil = 1 + \frac{12c_1+2d}{c_1+2c_2+7c_3+2d} < 2$ since c_1 is significantly smaller than c_3. This means that

Corollary 1. *When $m = 1$, we have that PKTAPP requires less transaction time than PKCROSS if $n = 1$ but more transaction time than PKCROSS if $n \geq 2$.*

In Proposition 2 we have shown that the number of application servers should be $\lceil \frac{(3c_1+2d)m+9c_1}{c_1+2c_2+7c_3+2d} \rceil$ more than the number of remote KDC realms so as to ensure that PKCROSS uses less transaction time than PKTAPP. But, the transaction time does not take into account the time required to wait in a queue for a service (i.e., waiting time) when multiple authentication requests present in any one of the authentication entities. Response time is used when such a case is considered. That is, *the response time* is the transaction time plus the waiting time per authentication request. A further discussion of the response time is given in next section.

2.2 The Calculation of Response Time via Queueing Networks

In a distributed system, a queueing network is an efficient tool to analyze system scalability. To investigate the scalability of PKTAPP and PKCROSS, we first model the entities, the client, the local KDC, the remote KDCs, the application servers and communication networks, as a queueing network. The client may represent a single or multiple users that request group authentication at a given rate and the authentication request is processed in these entities according to these two techniques. Figures 3 and 4 give queueing networks to describe the message flows of PKCROSS and PKTAPP where each system resource is modeled as a queue associated with a queueing discipline.

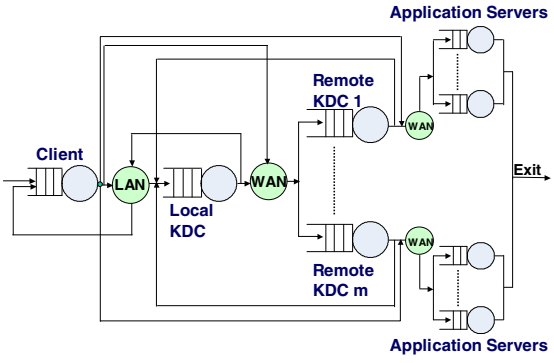


Fig. 3. A Queueing Network with m Remote Realms for PKCROSS

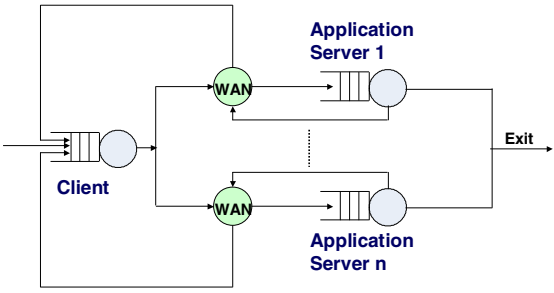


Fig. 4. A Queueing Network with n Application Servers for PKTAPP

Since a public key consumes a significantly higher computation cost than a private key, it is not reasonable to assume that all client requests are served at the same average service time. Instead, a class-switching in [7] and [28] is employed to model the class transaction with switching from low- to high-priority class, as different types of encryption/decryption operations are required with different service times. Preemption-resume is one good way to implement a service for satisfying multiple class client requests. In this paper, we use a *preemptive-resume* priority discipline, i.e., the service of a class r_2 request can be interrupted if a higher-priority request of class r_1 ($r_2 > r_1$) arrives during its service. The interrupted request resumes its service from where it stopped after the higher-priority request, and any other request with priority higher than r_2 that may arrive during its service, complete their service.

Our goal is to use the queueing networks to calculate the response time of an authentication request. The calculation is sketched below. Assume that the client requests group authentication at a rate λ. Based on the forced law, the throughput of an entity (the client station, the local KDC, the remote KDCs, or the application servers) is $X^{(j)} = \lambda v^{(j)}$ for class j job where $v^{(j)}$ is the number of visits to the entity by class j jobs. Then, the total throughput X is a sum of $X^{(j)}$ over all job classes at the entity. Thus, according to the utilization law, the utilization of the entity by class j jobs is $\rho^{(j)} = X^{(j)}\mu^{(j)} = \lambda v^{(j)}\mu^{(j)}$ where $\mu^{(j)}$ is the service time per visit to the entity by class j jobs. Hence, the total utilization ρ is a sum of $\rho^{(j)}$ over all classes at the entity. Thereby, the response time of the entity is $R = \frac{\mu}{1-\rho}$ where μ is an average service time of all classes at the entity, and the total response time per authentication request is a sum of vR over all entities.

To validate the accuracy of the queueing networks, we first simulated the scenario of these entities as shown in Figure 5 by doing the reference implementations of these two techniques under Windows XP. Moreover, a public-key cipher is usually associated with the calculations of 1024-bit precision numbers, so a public-key operation is computationally expensive and it costs as much as a factor of 1000 than an equivalent secret-key operations [13]. In the reference implementations we adopted the results from the Crypto++ ran on an Intel Core 2 1.83 GHz processor under Windows XP SP 2 in 32-bit mode [10]. Table 2 gives the time required to encrypt and decrypt a 64-byte block of data. Without loss of generality, we chose $c_1 = 0.000248$ msec, $c_2 = 0.07$ msec and $c_3 = 1.52$ msec. (Performance evaluation based on an ECC key will be discussed in

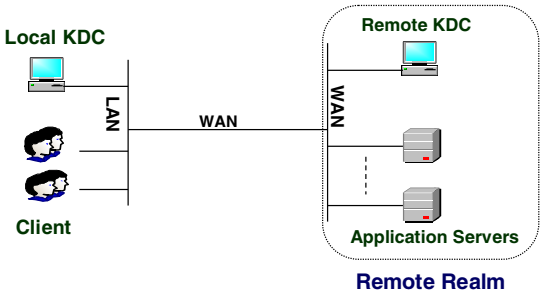


Fig. 5. A Test Scenario With a Remote Realm

Table 2. The computational times of Encryption and Decryption Operations

Protocols and Operation	Key Length	Computational times (msec)
AES/ECB	128	0.000248
AES/ECB	256	0.000312
RSA encryption	1024	0.07
RSA decryption	1024	1.52
RSA encryption	2048	0.15
RSA decryption	2048	5.95

Table 3. The Comparison of Analytic and Simulated Response Times

m=1		The Number of Application Servers				
Protocols	Methods	1	2	4	8	16
PKCROSS	Analytic	102.1	122.1	162.1	242.1	402.1
	Simulated	102.3	122.5	162.9	244.2	408.8
	R-Err%	-0.22	-0.30	-0.47	-0.85	-1.63
PKTAPP	Analytic	82.10	164.05	327.96	655.76	1311.38
	Simulated	82.23	164.56	329.97	663.87	1344.23
	R-Err%	-0.15	-0.30	-0.61	-1.22	-2.44

another paper.) Table 3 shows the accuracy of analytical response times obtained by the queueing methods compared to simulated results based on the scenario shown in Figure 5, where R-Err% is the relative error used to measure the accuracy of the analytic results compared to model simulation results, and it is defined by (analytic result - simulated result)/simulated result $\times$ 100. As seen in the table, the analytic response times match the simulated results very well.

To investigate performance with an increased number of application servers and remote realms, we further presented response times as a function of authentication request rates, i.e., throughput, when there are two remote realms, as shown in Figure 6. As is seen, PKCROSS has slightly less response time than PKTAPP when $n = 4$, called *a crossover number*. That is, PKCROSS is more efficient than PKTAPP if $n \geq 4$, but less efficient than PKTAPP if $n < 4$. Clearly, it follows from Table 3 that the crossover number is equal to 2 when $m = 1$. In general, Figure 7 shows crossover numbers with varying number of remote realms. The crossover numbers can be 99.8% perfectly fitted

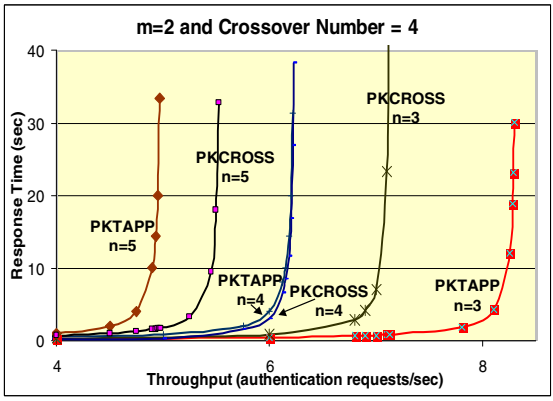


Fig. 6. Response Time vs. Authentication Request Rate

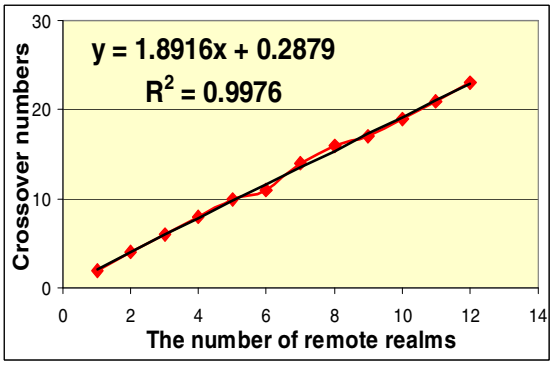


Fig. 7. Crossover Numbers vs. The Number of Remote Realms

by the straight line: $n = 1.8916m + 0.2879$. This means that PKCROSS is more efficient than PKTAPP when $n \geq \lceil 1.8916m + 0.2879 \rceil$. In other word, PKTAPP does not scale better than PKCROSS. That is different from what PKTAPP's designers expected.

3 Related Work

Kerberos (see RFC1510 [24]) has changed rapidly since 1999. Among them, there have been numerous proposals to integrate public key cryptography into Kerberos [9], [11], [14], [22], [16], [26], [29], [34], and [32]. These proposals address various concerns of Kerberos in distributed networks, for instance, security, scalability, and portability. Neuman, et al. [29] proposed PKINIT to enable use of public key cryptography for an initial authentication between the client and its local KDC. PKINIT is an extension of the Kerberos protocol. Its mechanism has been developed under the IETF Kerberos

WG for eleven years [4] before PKINIT was approved as an IETF Proposed Standard (RFC4556, see Zhu and Tung [34]).

PKCROSS [22] has been proposed to simplify the administrative burden of maintaining cross-realm keys so that it improves the scalability of Kerberos in large multi-realm networks. Public key cryptography takes place only KDC-to-KDC authentication. PKINIT and PKCROSS are centralized KDC protocols. Sirbu and Chuang [32] extended these two protocols to create PKDA for improving scalability and addressing the single point of failure on the KDC. PKTAPP is only a slight variation on the PKDA specification. Both PKDA and PKTAPP use lengthy public-key message exchanges between the client and the application servers, so they may not be anymore efficient than public key enabled authentication once with a KDC and faster secret-key cryptography for subsequent encryption with application servers [21]. PKTAPP is also known as KX.509/KCA [14] and Windows has its own protocol which is the equivalent to KX.509/KCA (see Altman [4]). Although the evolution of PKTAPP, its protocol structure has not been dramatically changed. We believe that PKCROSS and PKTAPP will revive soon. The Kerberos Consortium at MIT was just formed in September 2007 and listed PKCROSS as Project 10 in its proposal [27]. Kerberos on the web and Kerberos on mobile devices have been included in the strategic pillars of the MIT Kerberos Consortiym [18].

Performance evaluation is a fundamental consideration in the design of security protocols. However, performance associated with most of these protocols has been poorly understood. To analyze the performance of a protocol, we can first implement the protocol and then analyze measurement data taken from the implementation. But, the implementation of a protocol is very time-consuming and constricted to development resources and funding. While the implementation of KX.509 was released in 2007 [25], up to today, PKCROSS has not been implemented yet due to a matter of lack of development resources and funding [5]. A group of scientists have recently started to discuss the implementation of PKCROSS [19] and [20]. Hence, performance modeling has become an attractive approach since it can quickly provide a performance guidance used for a protocol design. The simplest modeling approach for a study of security protocols is to coount the number of secret, private, public key operations and then compute their corresponding costs (for example, see Amir et al. [2] and Steiner et al. [33]). This approach is straightforward. But, as shown in Sections 2, it is not easy to compute the number of operations for PKCROSS and PKTAPP since their message exchanges become complicated in the case of multiple KDC remote realms. Furthermore, this approach does not consider the waiting time of an authentication request in a queue. Hence, in order to efficiently and effectively analyze the performance of a protocol, we use a queueing network model as our protocol evaluation tool.

Queueing theory has been used in an analysis of chanllge/response authentication protocols in Liang and Wang [17]. But, they studied the performance of these protocols by only using a single $M/M/1$ queue which is the simplest case in queueing theory. Existing studies in Harbitter and Menasce [21] employed the performance modeling approach for examining the impact of PKCROSS and PKTAPP on network throughput based on their skeleton implementations and construction of *closed* queueing networks for a relatively simple case: there is only a single remote realm. This present paper

also employs a performance modeling approach for a study of PKCROSS and PKTAPP but extends Harbitter and Menasce [21] in several essential and important aspects. *First*, while the performance of PKCROSS and PKTAPP was studied in Harbitter and Menasce [21], it remains unknown which technique performs better in multiple remote realms that are typical in an increasingly large network these days. It is very difficult to analyze the case of multiple remote realms due to the complexity of authentication message exchanges. The difficulty is in the complexity analysis of these protocols and the building and analysis of queueing network models that reflects the workload of authentication requests for these protocols in the case of *multiple* remote realms. *Second*, we explicit derive the formulas for calculating the computational and communication times of these protocols so as to easily determine which technique is better. *Third*, using a closed queueing network in Harbitter and Menasce [21] assumes there exist *constant* authentication requests in the queueing network. This means that the number of the client's authentication requests remains unchanged with time, which is obviously an unrealistic assumption in a real-world computer application such as Web services. Rather, we adopted an *open* queueing network where the client requests authentication at a given rate, i.e., the number of authentication requests in a computing system under study is *not constant*; it can be dynamically changed with time. *Fourth*, in order to better understand the performance of the authentication technique, we distinguish the processing ordering of multiple authentication requests by using the preemptive-resume priority discipline.

4 Conclusions and Future Work

Public-key enabled Kerberos-based techniques such as PKINIT, PKCROSS, PKDA and PKTAPP (a.k.a KX.509/KCA) give a potential effective way for cross-realm authentication. However, the authentication problem is simple to describe but hard to solve, especially for multiple realms. Their performance has been poorly understood. In this paper we presented a throughout performance evaluation of PKCROSS and PKTAPP in terms of computational and communication times, and through the use of validated analytical queueing models. Our analysis revealed that PKTAPP does not perform better than PKCROSS in a large network where there are many application servers within either a single or multiple remote realms.

As is shown, our performance methodology based on complexity analysis and queueing theory is an effective way to analyze the performance of security protocols. In the future, we plan to apply the methodology to other protocols, such as IKEv2 in Kaufman [23] and AAA in Patel et al. [30]. Particularly, it is of very interest to apply our method to those protocols which are used in either time-sensitive or resource-limited applications, e.g., the ones in wireless sensor networks.

Acknowledgments. This work is supported in part by the U.S. National Science Foundation (NSF) under CNS 1065665 and NSF/BBN under #1895 through CNS 1128122. The contents of this paper do no necessarily reflect the position or the policies of the U.S. Government.

References

1. Amir, Y., Kim, Y., Nita-Rotaru, C., Tsudik, G.: On the performance of group key agreement protocols. ACM Transactions on Information and Systems Security (TISSEC) 7(3), 1–32 (2004)
2. Amir, Y., Kim, Y., Nita-Rotaru, C., Schultz, J., Stanton, J., Tsudik, G.: Secure group communication using robust contributory key agreement. IEEE Transactions on Parallel and Distributed Systems 15(5), 468–480 (2004)
3. Al-Janabi, S.: Public-Key Cryptography Enabled Kerberos Authentication. In: Developments in E-systems Engineering, DeSE (2011)
4. Altman, J.: NIST PKI 2006: Integrating PKI and Kerberos (2007), http://www.secure-endpoints.com/talks/nist-pki-06-kerberos.pdf
5. Altman, J.: Personal communication (2007)
6. Barry, D.: Web Services and Service-Oriented Architecture: Your Road Map to Emerging IT. Morgan Kaufmann (2003)
7. Bruell, S., Balbo, G.: Computerational Algorithms for Closed Queueing Netowrks. In: Denning, P.J. (ed.) Science Library. Elsevier North Holland, Inc., New York (1980)
8. Buckley, S.: MIT Kerberos Consortium Proposal to Sponsors (2008), http://www.kerberos.org/join/overview.pdf
9. CITI. kx509 and KCA (2006), http://www.citi.umich.edu/projects/kerb_pki/
10. Dai, W.: Crypto++ 3.1 benchmarks (2007), http://www.eskimo.com/~weidai/benchmark.html
11. Davis, D.: Kerberos plus RSA for world wide web security. In: Proceedings of the First USENIX UNIX Workshop on Electronic Commerce, New York City, New York (July 1995)
12. Davis, D.: Compliance defects in public-key cryptography. In: Proceedings of the Sixth USENIX UNIX Security Symposium (USENIX Security 1996), San Jose, California (July 1996)
13. Dongara, P., Vijaykumar, T.N.: Accelerating private-key cryptography via multithreading on symmetric multiprocessors. In: Proc. IEEE Int'l Symp. Performance Analysis of Systems and Software (ISPASS 2003), pp. 58–69. IEEE Press (2003)
14. Doster, W., Watts, M., Hyde, D.: The KX.509 Protocol (2001), http://www.citi.umich.edu/techreports/reports/citi-tr-01-2.pdf
15. Garman, J.: Kerberos: The Definitive Guide. O'Reilly (2003)
16. Kirsal, Y., Gemikonakli, O.: Further Improvements to the Kerberos Timed Authentication Protocol. In: Sobh, T., Elleithy, K., Mahmood, A., Karim, M. (eds.) Novel Algorithms and Techniques In Telecommunications, Automation and Industrial Electronics. Springer (2008)
17. Liang, W., Wang, W.: A Quantitative study of authentication and QoS in Wireless IP Networks. In: Proceedings of the 24th IEEE Conference on Computer Communications, INFOCOM (2005)
18. Hardjono, T.: Kerberos on the Web: Update. MIT Kerberos Consortium (December 2005), http://www.kerberos.org/events/Board-3-30-09/3-hardjono-kerbweb.pdf
19. Heimdal. PKCROSS for Heimdal (April 2008), http://www.taca.jp/krb-cross-realm/pkcross-heimdal.html
20. Heimdal. Initial version of PKCROSS Implementation. Heimdal Discussion Mailing List (April 2008), http://www.stacken.kth.se/lists/heimdal-discuss/2008-04/msg00004.html

21. Harbitter, A., Menasce, D.: Perofrmance of public-key-enabled Kerberos authentication in large networks. In: Proceedings of 2001 IEEE Symposium on Security and Privacy, Oakland, California (2001)
22. Hur, M., Tung, B., Ryutov, T., Neuman, C., Medvinsky, A., Tsudik, G., Sommerfeld, B.: Public key cryptography for cross-realm authentication in Kerberos (PKCROSS) (May 2001), http://tools.ietf.org/html/draft-ietf-cat-kerberos-pk-cross-07
23. Kaufman, C.: Internet Key Exchange (IKEv2) Protocol (December 2005), http://www.ietf.org/rfc/rfc4306.txt
24. Kohl, J., Neuman, C.: RFC 1510: The Kerberos network authentication service, v5 (1993), http://rfc.net/rfc1510.html
25. KX.509. KX.509 Source (2007), http://kx509.cvs.sourceforge.net/kx509/
26. Medvinsky, A., Hur, M., Neuman, C.: Public key utilizing tickets for application servers (PKTAPP) (January 1997), http://tools.ietf.org/html/draft-ietf-cat-pktapp-00
27. The MIT Kerberos Consortium. Proposal for corporate sponsors (2007), http://www.kerberos.org/join/proposal.pdf
28. Muntz, R., Chandy, K., Baskett, F., Palacios, F.: Open, closed, and mixed networks of queues with different classes of customers. Journal of the ACM (April 1975)
29. Neuman, B., Tung, B., Way, J., Trostle, J.: Public key cryptography for initial authentication in Kerberos servers (PKINIT 2002) (October 2002), http://ietf.org/internet-drafts/draft-ietf-cat-Kerberos-pk-init-02.txt
30. Patel, A., Leung, K., Khalil, M., Akhtar, H.: Authentication protocol for mobile IPv6 (2006), http://www.rfc-editor.org/rfc/rfc4285.txt
31. Pathan, K., Deshmukh, S., Deshmukh, R.: Kerberos Authentication System?A Public Key Extension. International Journal of Recent Trends in Engineering (May 2009)
32. Sirbu, M., Chuang, J.: Distributed authentication in Kerberos using public key cryptography. In: IEEE Symposium On Network and Distributed System Security, NDSS 1997 (1997)
33. Steiner, M., Tsudik, G., Waidner, M.: Diffie-Hellman key distribution extended to group communication. In: Proceedings of the 3rd ACM Conference on Computer and Communications Security, CCS 1996 (1996)
34. Zhu, L., Tung, B.: RFC 4556: Public key cryptography for initial authentication in Kerberos (PKINIT) (June 2006), http://www.ietf.org/rfc/rfc4556.txt

Boardroom Voting Scheme with Unconditionally Secret Ballots Based on DC-Net*

Long-Hai Li, Cheng-Qiang Huang, and Shao-Feng Fu

School of Computer Science and Technology
Xidian University, Xi'an 710071, China
{lhli,fsf}@xidian.edu.cn, cq.huang@stu.xidian.edu.cn

Abstract. A novel electronic voting scheme is proposed which is quite suitable for small scale election settings. An outstanding characteristic of the design is its guarantee of unconditionally perfect ballot secrecy. It satisfies self-tallying, fairness and verifiability. Disruption of the result of an election equals to breaking the Discrete Logarithm Problem. Our scheme is built on top of the DC-net(dining cryptographers network) anonymous broadcast protocol. It needs no trusted authority to guarantee its security, but assumes a complete network of secure private channels between voters.

Keywords: cryptographic protocol, electronic voting, anonymous broadcast, ballot secrecy, zero knowledge proof.

1 Introduction

A boardroom voting refers to a small-scale election activity which could take place in a boardroom or parliament involving a small number of participants. In large-scale electronic voting schemes(countrywide votings), voter privacy is usually compromised in favor of efficiency. When a large amount of ballots have been collected by a tallying center, it is expected to decrypt all the ballots and then publish the tallying result after a short delay. While in small-scale electronic voting schemes, people pay more attention to the secrecy of ballots. The number of voters is small, which indicates every ballot is critical to the tallying result. Thus, to ensure fairness, it is crucial to make sure that every voter's choice is not revealed.

In [1], Kiayias and Yung introduced the notion of *Perfect Ballot Secrecy* for the first time. And they also designed a boardroom electronic voting scheme with this feature. Perfect Ballot Secrecy(PBS) means that if someone wants to know the choice of a specific voter, he needs to collude with all the other voters. In other words, an attacker can not gain any useful information about an honest voter's choice through election itself except for the tallying result

* This work was supported by National Natural Science Foundation of China under grant 61101142 and the Fundamental Research Funds for the Central Universities under grant K50510030012.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 220–232, 2012.

and choices of dishonest voters. After [1], some more efficient electronic voting schemes with Perfect Ballot Secrecy were proposed, such as [2] [3] [4]. However, the PBS in these schemes is derived from the difficulty of some well-known problems, which means that they are just secure in the sense of computation. If some powerful computing devices with exponential computing capability were produced someday, such as the quantum computers and the biocomputers which have aroused world wide attention, these voting schemes would be insecure. And the filed ballots of past elections, although encrypted with some public-key cryptography algorithms, would be at stake.

To address this problem, an electronic voting scheme with Unconditionally Perfect Ballot Secrecy is designed by use of the DC-net(Dining Cryptographer Networks) anonymous broadcast protocol [5]. It can achieve Information-Theoretic ballot secrecy. Although our scheme depends on secret channels among participants, we argue that it is feasible to establish these channels in small-scale election settings.

The scheme needs no trusted authority to guarantee ballot secrecy, and satisfies self-tallying, fairness and verifiability. An adversary has to solve the Discrete Logarithm Problem to disrupt the integrity of elections. In addition, the basic single-candidate voting protocol is extended to cater for multiple candidates. A veto voting scheme based on DC-net with similar properties is also proposed.

2 Related Work

In recent decades, numerous electronic voting schemes based on cryptography technology have been proposed. As we know, they can be divided into three main classes: (1) voting schemes based on homomorphic encryption, such as [6] [7]; (2) voting schemes based on Mix-net [8] system, such as [9] [10]; (3) voting schemes based on blind signature, such as [11] [12]. Blind signatures are commonly used to ensure the eligibility of voters and the validity of ballots. Strictly speaking, only partial anonymity of voters can be achieved by using blind signatures. In order to achieve full anonymity, an anonymous communication tool, such as Mix-net or DC-net, must be involved in the voting system. Thus, a majority of electronic voting schemes based on blind signatures can be classified into the second category. Most of these schemes are used in large-scale election activities, and need one or more trusted authorities to collect, check, decrypt and then tally the ballots.

Boardroom electronic voting schemes are suitable for small-scale election activities. They are commonly designed in a decentralized fashion and need no trusted authority. They usually provide better guarantees of ballot secrecy, even though sacrificing some efficiency. Kiayias and Yung first introduced the notion of Perfect Ballot Secrecy in [1] and designed a boardroom voting scheme with this property. Their scheme satisfies self-tallying, which means everyone can easily calculate the voting result without the help of any trusted authority. After that, Damgard and Jurik [2], Groth [3], and Hao [4] proposed three boardroom voting schemes respectively which have similar attributes and are more efficient

than the original work [1]. However, Perfect Ballot Secrecy of these four schemes are based on the assumption of some well-known difficult problems, which means that they are only secure in the sense of computation.

Dining Cryptographers Network [5], abbreviated as DC-net, is an anonymous broadcast protocol first proposed by Chaum. It has no trusted authority, and takes full advantage of cooperation and mutual protection among participants to achieve anonymity. If there exist absolutely secure secret channels among participants, DC-net can achieve unconditional sender anonymity. This remarkable characteristic makes it quite suitable for hiding relationships between ballots and voters in small-scale election settings. For this reason, we designed a boardroom voting scheme with Unconditional Perfect Ballot Secrecy by using DC-net protocol. This scheme can provide information-theoretic perfect ballot secrecy. Although it relies on secret channels among participants, we consider it feasible to establish these channels in small-scale elections. For instance, two voters can exchange huge amount of data as shared secret keys in advance in a non-cryptographic way, such as by sending a DVD to each other via postal services, so as to use them for encryption in elections afterwards.

Only a handful of DC-net based voting schemes can be found in the literature, including [13] and [14]. While achieving unconditional secrecy as well, they just use DC-net system as a tool of anonymous communication. In [13], if there are n participants, DC-net protocol needs to be operated for n times at least, which means that they are practically inefficient. However, our scheme skillfully utilizes the additivity property of DC-net, and carries out a voting with n voters by a small number of DC-net operations. All ballots are added together automatically, achieving self-tallying naturally. Thus it is significantly more efficient than [13] and [14] in terms of rounds, computational cost and bandwidth usage. Similar to [1] and [3], our scheme needs a trusted authority only when we need to achieve fairness. Yet the secrecy and integrity don't rely on the trusted authority.

3 Voting Scheme with Unconditionally Ballot Secrecy

3.1 Security Requirements

The requirements we want the boardroom voting protocol to fulfill are the following.

Perfect Ballot Secrecy. This requirement says an attacker needs to collude with all remaining voters to get the voting choice of a specific voter (of course this voter doesn't collude with the attacker). In other words, even though an attacker obtains all public transcripts including tallying result output by a voting protocol and the secret inputs including voting choices of dishonest voters, he still can not gain any useful information about an honest voter's secret choice anyhow. This is the best type of anonymity we hope for in elections. And our scheme provides a stronger protection for ballot secrecy because we can still guarantee Perfect Ballot Secrecy even though attackers have infinite computational capability.

Integrity. This requirement says any voter is not able to pose more impact on election result beyond the ballot of his own; any voter can not disrupt election result without being catched.

Self-tallying. Anybody is possible to calculate the tallying result without the help of any third trusted authority after all voters have cast their votes.

Verifiability. Any voter is able to verify the correctness of the tallying result and check whether other voters acted honestly.

Fairness. Nobody can obtain a partial tally before the deadline. Otherwise a voter who is left behind in casting his ballot and has learned a partial result would change his or her choice according to the situation.

3.2 Protocol Details

Notations and Assumptions. We assume in the following that notation $P = \{P_1, \ldots, P_n\}$ represents the set of voters. G stands for a finite cyclic group where the Discrete Logarithm problem is hard. $q = |G|$ is a big prime number and g, h are two generators chosen randomly from G. Assume also that there exists an absolutely secure channel between any two voters, which is used to negotiate session keys. In addition, we assume a reliable broadcasting system. If the network infrastructure does not physically ensure reliable broadcasting, we can introduce a reliable Web BBS or implement the broadcast channel over a asynchronous peer-to-peer communication network by utilizing Byzantine Agreement techniques [15] [16]. To ensure fairness, we will regard BBS as the (n+1)th voter and also make it the last one to broadcast throughout the voting process. Here the fairness demand is fulfilled in a relaxed way such that it is guaranteed by a hopefully honest authority, i.e. BBS.

The whole protocol consists of 2 rounds of DC-net protocol and 3 rounds of broadcasting. That means every voter needs to participate in 5 rounds of broadcasting in total. Among the 5 rounds of broadcasting, the 1st round is intended to make sure the secret session keys owned by all voters satisfy some desirable properties; the 2nd round is used by voters to cast their votes; the remaining rounds are employed to check the correctness of tallying result.

For easy exposition, we first present a yes/no voting where voters vote for a proposal or a single candidate. The content of a ballot is 1 or -1. 1 means agreement and -1 means disagreement. Thus voting result equals to the discrepancy between approvers and opponents. Apparently, this method can also be used in elections which just include two candidates.

1st round of broadcasting: Verifying secret session keys.

- Every two voters, denoted by P_i and P_j respectively, negotiate two shared secret keys $t_{ij} = t_{ji} \in \mathbb{Z}_q$ and $k_{ij} = k_{ji} \in \mathbb{Z}_q$ with each other through the secret channel between them. It is possible for this negotiation process to take place before the beginning of an election.

- When the key negotiation process has been done, each voter P_i calculates his session keys as follows:

$$k_i = \sum_{j=1}^{n+1} \mathrm{sgn}(i-j)k_{ij} \ , \ t_i = \sum_{j=1}^{n+1} \mathrm{sgn}(i-j)t_{ij} \ .$$

sgn in the above equation denotes the sign function which is defined as follows:

$$\mathrm{sgn}(x) := \begin{cases} -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases} .$$

Then P_i chooses $u_i, c_i \in_R \mathbb{Z}_q$, where c_i will be used later to generate the challenge for the interactive proof protocol ensuring validation of ballots.

- Each voter P_i broadcasts the following messages:

$$V_{1,i} = g^{k_i} h^{t_i} \ , \ R_i = g^{c_i} h^{u_i} \ .$$

$V_{1,i}$ can be regarded as the Pederson commitment [17] for secret key k_i, and t_i is the secret commitment key. R_i can be regarded as the Pederson commitment for c_i, and u_i is the secret commitment key.

- After all the voters have done broadcasting, any voter can calculate and verify that:

$$V_1 = \prod_{i=1}^{n+1} V_{1,i} = 1 \ .$$

If $V_1 \neq 1$, it means that someone must have cheated. Thus, we can force all voters to publish their secret session keys in order to find out who has cheated. After all cheaters have been expelled, the remaining voters restart the protocol. If a disputing between two voters occurs during this period, those two voters will be forbade to negotiate shared secret keys for the next run. This is equivalent to removing an edge between two vertexes from the key graph of DC-net. In this way, a cheater will be eliminated from the system if he has triggered controversy more than n times.

2nd round of broadcasting: Voting.
The substantial voting process is involved in this round. Suppose that $b_i \in \{1, -1\}$ is the content of the ballot of P_i, and that b_{n+1} of P_{n+1}, i.e.BBS, is 0. Then P_i broadcasts the following value:

$$V_{2,i} = k_i + b_i \ .$$

After all voters have done voting, anyone can calculate the tallying result like this:

$$V_2 = \sum_{i=1}^{n+1} V_{2,i} \ .$$

We will accept this result only if all ballots can successfully pass the verification program discussed afterwards.

Prover(P_i)			Verifier
Public Parameter: $Y_i = V_{1,i}g^{-V_{2,i}} = g^{-b_i}h^{t_i}$			
$b_i = 1$	$b_i = -1$		
$\alpha_i, \beta_i, d_{2,i} \in \mathbb{Z}_q$,	$\alpha_i, \beta_i, d_{1,i} \in \mathbb{Z}_q$,		
$A_i = h^{\alpha_i}$,	$A_i = (Y_i g)^{-d_{1,i}} h^{\alpha_i}$,	$\xrightarrow{A_i, B_i}$	
$B_i = (Y_i/g)^{-d_{2,i}} h^{\beta_i}$	$B_i = h^{\beta_i}$		
		$\xleftarrow{c=c_1+\cdots+c_{n+1}}$	$c \in_R \mathbb{Z}_q$
$d_{1,i} = c - d_{2,i}$,	$d_{2,i} = c - d_{1,i}$,		$c =_? d_{1,i} + d_{2,i}$
$r_{1,i} = \alpha_i + d_{1,i}t_i$,	$r_{1,i} = \alpha_i$,	$\xrightarrow{r_{1,i}, r_{2,i}, d_{1,i}, d_{2,i}}$	$h^{r_{1,i}} =_? A_i(Y_i g)^{d_{1,i}}$
$r_{2,i} = \beta_i$	$r_{2,i} = \beta_i + d_{2,i}t_i$		$h^{r_{2,i}} =_? B_i(Y_i/g)^{d_{2,i}}$

Fig. 1. Proving that $(V_{1,i}g^{-V_{2,i}} = gh^{t_i}) \bigvee (V_{1,i}g^{-V_{2,i}} = g^{-1}h^{t_i})$

3rd, 4th and 5th rounds of broadcasting: Checking ballots.
In the last three rounds of broadcasting, we execute an interactive proof protocol to ensure that the ballots voted in round 2 are valid. See Figure 1 for detail.In the 3rd round, P_i calculates and broadcasts A_i and B_i as it is depicted in Figure 1.

In the 4th round, P_i broadcasts his c_i which have been generated in round 1. When all the voters have done broadcasting, $c = c_1 + c_2 + \cdots + c_{n+1}$ will be calculated as a challenge for the interactive proof protocol depicted in Figure 1.

In the 5th round, P_i calculates and broadcasts the following values based on c: $d_{1,i}, d_{2,i}, r_{1,i}, r_{2,i}$ and u_i. With this, all voters can verify the validity of P_i's outputs according to Figure 1. And they should also verify that the equation, $R_i = g^{c_i}h^{u_i}$, holds true.

If someone can not pass the verification program in round 5 or refuses to participate in the proof protocol, he will be regarded as a cheater and expelled from the system. If this case happened, the protocol would be executed again. While, if all voters have passed checking, we believe that the voting result in round 2 is valid and terminate the protocol successfully.

Note that proof protocol for the special participant P_{n+1} is different from the one used above, however its basic idea is similar. We omit it due to limited space.

4 Analysis

4.1 Security Analysis

In this section, we analyze and justify the security properties the above protocol satisfies.

Self-tallying. The 1st round of our protocol which deals with shared secret key negotiation and verification ensures that the secret keys produced by $n+1$ voters satisfy the following equations:

$$\sum_{i=1}^{n+1} k_i = 0 \ , \ \sum_{i=1}^{n+1} t_i = 0 \ .$$

Consequently, if all voters have constructed ballots $V_{2,i}$ by following the protocol strictly, then the equation below holds:

$$V_2 = \sum_{i=1}^{n+1} V_{2,i} = \sum_{i=1}^{n+1} k_i + \sum_{i=1}^{n+1} b_i = \sum_{i=1}^{n+1} b_i \ .$$

Unconditionally Perfect Ballot Secrecy

Theorem 1. If there exists two or more honest voters in an election following the described protocol, then except for the tallying result, any adversary cannot learn any useful information about the ballot voted by an honest voter even the adversary has unbounded computing power.

Proof(schech): We prove this theorem holds true by demonstrating that all transcripts output by the protocol are probabilistically distributed independently of the ballots voted by honest voters.

- Recall that the two secret session keys owned by voter P_i are: $k_i = \sum_j sgn(i-j)k_{ij}$, $t_i = \sum_j sgn(i-j)t_{ij}$. As long as one of the other voters chooses his shared secret keys k_{ij} and t_{ij} randomly and honestly, it can be sure that k_i and t_i are uniformly distributed over $\mathbb{Z}_q$. Therefore, to reveal k_i and t_i, an attacker needs to control all the other voters, which is in contradiction with the assumption that there are two or more honest voters. Further more, since t_i is random, the commitment value $V_{1,i}$ is uniformly distributed over G, which means that it won't reveal any information about secret key k_i. Because k_i is random, $V_{2,i} = k_i + b_i$ is uniformly distributed over $\mathbb{Z}_q$, which means it won't expose any information of ballot b_i. Thus, an attacker can gain nothing about the content of a ballot through the output of the first two rounds of broadcasting, even though he may have infinite computational capability.
- The interactive proof protocol of Figure 1 is Special Honest Verifier Zero Knowledge [18] [1]. This is in that given a random challenge c, if we choose at random $\alpha_i, \beta_i, d_{1,i}, d_{2,i}$ from $\mathbb{Z}_q$ s.t. $c = d_{1,i} + d_{2,i}$, the conversation $(A_i = (Y_i g)^{-d_{1,i}} h^{\alpha_i}$, $B_i = (Y_i/g)^{-d_{2,i}} h^{\beta_i}$, c, $r_{1,i}$, $r_{2,i}$, $d_{1,i}$, $d_{2,i})$ is an accepting conversation that has the same distributions as those of the accepting conversations generated by a prover and an honest verifier. Even with unlimited computational capability, an adversary can't make a distinction between the above two cases. The fact that such conversation simulations

exist indicates that the transcript of a run of the proof protocol between the prover P_i and some (possibly dishonest) verifier is distributed independently of the witness (b_i, t_i). Therefore, the last three rounds of broadcasting which implement the interactive proof protocol do not leak any Shannon information about the ballots of honest voters. □

Integrity

Theorem 2. For any voter, the probability that he votes an invalid ballot without being detected in our protocol is negligible unless he is able to solve the Discrete Logarithm Problem.

Proof(schech):

- During the 1st round of broadcasting, if a dishonest voter P_i did not obey the protocol in constructing k_i and t_i, the probability that $\prod V_{1,i} = 1$ would be negligible. The protocol would enter into an investigation program with all voters being forced to publish their secret keys. To avoid being regarded as a cheater, P_i has to fabricate a dispute to share the blame with others. When the protocol restarts, two voters involved in a dispute will be forbade to negotiate shared keys again. This is analogous to removing an edge between two vertexes from the key graph of DC-net. After n times of disputes at most, P_i will be identified as a cheater and expelled from the system.
- In the 2nd round of broadcasting, let's suppose that P_i has not constructed ballot $V_{2,i}$ following the protocol. There exist just three possible cases in which P_i can pass the verification of the 5th round: (1) P_i uses a different key k_i', not k_i, to encrypt b_i and P_i knows the $t_i' \neq t_i$ which satisfies that $g^{k_i'} h^{t_i'} = V_{1,i} = g^{k_i} h^{t_i}$. However, this is in contradiction with the assumption about the difficulty of Discrete Logarithm problems. (2) P_i guesses correctly the challenge c generated by all voters in round 4 cooperatively. This probability is just $1/q$. (3) In round 4, P_i manipulates the probability distribution of the challenge c by broadcasting a proper c_i' to supersede c_i in the end. But P_i needs to know $u_i' \neq u_i$ which satisfies $g^{c_i'} h^{u_i'} = R_i = g^{c_i} h^{u_i}$. We know that this is in contradiction to the DLP assumption as well. So, P_i will be identified in round 5 if he has cheated in round 2 with an overwhelming probability. And the soundness of the proof protocol depicted by Figure 1 also ensures that b_i is either 1 or -1. As a result, P_i can not pose more impact on voting result V_2 beyond the ballot b_i of his own.
- The correctness and soundness of the proof protocol in Figure 1 ensure that a dishonest voter who cheats in round 3 or 5 will be unmasked with an overwhelming probability. And in round 4, a dishonest voter needs to solve the Discrete Logarithm Problem to publish a challenge value which is different from c_i chosen in round 1.

For these reasons, under the assumption of Discrete Logarithm Problem, the probability that a voter who cheats in the protocol is not catched by others is negligible. □

Compared to voting schemes in [1] and [3], our scheme provides comparatively weak fault tolerance. That is, when someone cheats in our protocol or quits ahead of the end, we can just start over, not being able to recover form the error and proceed by remaining members. In fact, this is an intrinsic problem in using secret channels among participants. Even so, the cost of restarting our protocol is acceptable in small-scale elections. And, by the way of punishing cheaters heavily, we can, in some extend, avoid the protocol being operated repeatedly.

Verifiability. BBS system ensures that every participant can see all the outputs of the protocol. And the contents seen by them are the same. Every voter can ensure the correctness of session keys of others by checking $\prod_{i=1}^{n+1} V_{1,i} = 1$; every voter can ensure that every ballot is valid by taking part in the process of interactive proof in round 3, 4 and 5, thus further ensure that the voting result V_2 in round 2 is correct.

It should be pointed out that we achieve just internal verifiability. That is only voters who participate in online interactive proof can verify the correctness of voting result. The outputs of round 3, 4 and 5 are meaningless for external observers, because all voters can collude together to construct apparently valid voting transcripts.

To enable an observer to verify voting result, that is to say to achieve public verifiability, we can transform the proof protocol in Figure 1 into a non-interactive proof protocol by using Fiat-Shamir techniques [19]. In that case, every voter is required to broadcast a proof of validity as well as his ballot $V_{2,i}$ during round 2. We have to introduce the Random Oracle Model [20] when analyzing the security of this modified scheme.

Fairness. Before P_{n+1} (i.e. BBS) casts his ballot, the other n actual voters can not learn the tallying result in advance because all ballots must be summed up to render the session keys $k_1, k_2, \ldots, k_{n+1}$ canceled out mutually. So the fairness depends on the honestness of BBS.

Table 1. Comparison with past work

Protocols	Round	Exp for ballot	Exp for Proof	Exp for Verification
Kiayias-Yung [1]	3	$2n+2$	$3n+5$	n^2+6n-7
Groth [3]	$n+1$	4	7	$7n$
Hao-Ryan-Zielinski [4]	2	2	5	$5n-5$
–	5	2	2	$3n$

4.2 Efficiency Analysis

We don't take into account the cost of negotiating shared keys through secret channels due to the difficulty of estimating. Suppose that n represents the number of voters, l is the security parameter used in generating group G.

Apparently, it requires 4 modular exponentiations to generate $V_{1,i} = g^{k_i} h^{t_i}$ and $R_i = g^{c_i} h^{u_i}$ in round 1 for each voter. In fact, we can halve the number of exponentiations required to generate $V_{1,i}$ and R_i by using the algorithm for product of exponentiations proposed in [21]. The product of two exponentiations in the formulation of $g^x h^y$ can be computed at the cost of a single exponentiation using this algorithm. In a similar way, it requires 2 exponentiations to compute A_i and B_i for each voter in round 3. In round 5, verifying the proof for the validity of P_i's ballot incurs 3 exponentiations (including 1 exponentiation to verify the commitment of c_i). So, every voter needs to compute $3n$ modular exponentiations in total to verify the validity of the voting result. The computational cost in round 2 and 4 is negligible. Overall, every voter needs to compute $3n + 4$ modular exponentiations and broadcasts $\mathbf{O}(l)$ bits of data throughout a run of the protocol.

The numbers of exponentiations required by 4 decentralized voting protocols with similar security properties are listed in Table 1. The comparison shows that our protocol is more efficient than [1] [3] and [4] in terms of computational cost. But it requires more rounds of public discussion than [4]. Our scheme also surpasses [13] and [14] as the computation complexity of them is $\mathbf{O}(n^2)$.

5 1-Out-of-n Voting-Scheme

The above scheme is appropriate for yes/no elections. Now we will elaborate how to extend it to the situations in which multiple candidates are involved.

The first method is using the techniques in [6] and [7]. Assume that there are m candidates. We choose at random m generators, $f_1, f_2, \ldots, f_m$, from G. The process of negotiating and verifying session keys remains unchanged. In the 2nd round of broadcasting, if P_i chooses candidate $b_i \in \{1, 2, \ldots, m\}$, then he broadcasts $V_{2,i} = g^{k_i} f_{b_i}$. In round 3, 4 and 5, P_i is required to prove that he knows the secret keys k_i and t_i which satisfy the following equations:

$$V_{1,i} = g^{k_i} h^{t_i} \bigwedge V_{2,i} = g^{k_i} f_{b_i} \bigwedge b_i \in \{1, 2, \ldots, m\} \ .$$

The corresponding interactive proof protocol is similar to Figure 1. Curious readers can refer to [18] for more details.

To obtain the ballots for each candidate, we calculate the following value:

$$V_2 = \prod_i V_{1,i} = g^{\sum_{i=1}^{n} k_i} f_1^{v_1} f_2^{v_2} \cdots f_m^{v_m} = f_1^{v_1} f_2^{v_2} \cdots f_m^{v_m} \ .$$

Note that $v_1, v_2, \ldots, v_m$ are less than n, hence we can get $v_1, v_2, \ldots, v_m$, which represent the ballots of each candidate respectively, by brute force search.

If the number of voters n and that of candidates m satisfy $m\lceil \log_2 n \rceil \leq \lfloor \log_2 q \rfloor$, we can use a more efficient method to avoid brute force search. Let $e = \lceil \log_2 n \rceil$. P_i broadcasts

$$V_{2,i} = k_i + 2^{e(b_i - 1)}$$

```
for i = 1 to m
        v_i = V_2  mod 2^e, output v_i
        V_2 = V_2 ≫ e        // shifting V_2 right for e bits
end of for
```

Fig. 2. Counting algorithm

in round 2 if he chooses candidate b_i. Apparently, $V_{2,i} \in \mathbb{Z}_q$, so the length of P_i's output is $\log_2 q$ bits. In essence, this method divides $\log_2 q$ bits into m sub-channels, and each one containing e bits is corresponding to a candidate. To vote for a specific candidate, P_i sends 1 into the corresponding sub-channel and sends 0 into the other sub-channels. He also needs to prove that he knows k_i and t_i in round 3, 4 and 5 , and k_i, t_i satisfy:

$$V_{1,i} = g^{k_i} h^{t_i} \bigwedge g^{V_{2,i}} = g^{k_i} g^{2^{e(b_i-1)}} \bigwedge b_i \in \{1, 2, \ldots, m\} \ .$$

Suppose that $V_2 = \sum_i V_{2,i}$, the new algorithm for tallying ballots is illustrated in Figure 2.

Note that $v_1, v_2, \ldots, v_m$ outputted by the algorithm above are the corresponding ballots for m candidates.

The second method is more efficient than the first one, [6] and [7], because it involves only addition operations in the 2nd round of broadcasting and tallying process.

6 Anonymous Veto Voting-Scheme

In veto voting settings, as long as one voter submits a negative vote, the proposal discussed will be rejected. We now transform the basic scheme in section 3 into a veto voting scheme.

The process of generating and verifying session keys in round 1 remains unchanged. In round 2, if P_i wants to veto the proposal, he chooses $r \in_R \mathbb{Z}_q^*$ and outputs $V_{2,i} = k_i + r$. If he wants to accept it, he should output $V_{2,i} = k_i$. After the 2nd round of broadcasting, we can calculate $V_2 = \sum_i V_{2,i}$. Then if $V_2 \neq 0$, it means that someone vetoed. If $V_2 = 0$, it means that all the voters agree with the proposal. And if someone didn't construct $V_{2,i}$ correctly, it would have the same effect as the case in which someone vetoed. Hence, we don't need round 3, 4 and 5 to carry out interactive proofs.

The probability that this scheme fails to operate successfully is equal to the probability that the sum of all random r submitted by voters is q. Therefore the probability of failing is $1/q$ and negligible.

It's easy to prove that this scheme provides unconditionally perfect ballot secrecy as well.

7 Conclusion

In this paper, we have designed a boardroom electronic voting scheme using DC-net protocol under the assumption of untappable secret channles. The most out-

standing characteristic of this scheme is that it provides Information-Theoretically perfect ballot secrecy. It offers the function of self-tallying, and satisfies fairness and verifiability. The integrity of voting result relies on the difficulty of solving Discrete Logarithm problem. We have also proposed an 1-out-of-n voting-scheme and a veto voting-scheme by extending the basic yes/no voting scheme. Our voting scheme is more efficient than previous schemes with similar security properties including [1] [2] [3] [4] and [13] in terms of computational cost.

References

1. Kiayias, A., Yung, M.: Self-tallying Elections and Perfect Ballot Secrecy. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 141–158. Springer, Heidelberg (2002)
2. Damgard, I., Jurik, M.: A Length-flexible Threshold Cryptosystem with Applications. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, Springer, Heidelberg (2003)
3. Groth, J.: Efficient Maximal Privacy in Boardroom Voting and Anonymous Broadcast. In: Juels, A. (ed.) FC 2004. LNCS, vol. 3110, pp. 90–104. Springer, Heidelberg (2004)
4. Hao, F., Ryan, P., Zielinski, P.: Anonymous Voting by 2-Round Public Discussion. IET Information Security 4(2), 62–67 (2010)
5. Chaum, D.: The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability. Journal of Cryptography 1(1), 65–75 (1988)
6. Cramer, R., Franklin, M.K., Schoenmakers, B., Yung, M.: Multi-authority Secret-Ballot Elections with Linear Work. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 72–83. Springer, Heidelberg (1996)
7. Cramer, R., Gennaro, R., Schoenmakers, B.: A Secure and Optimally Efficient Multi-authority Election Scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997)
8. Chaum, D.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM 24(2), 84–88 (1981)
9. Fujioka, A., Okamoto, T., Ohta, K.: A Practical Secret Voting Scheme for Large Scale Elections. In: Zheng, Y., Seberry, J. (eds.) AUSCRYPT 1992. LNCS, vol. 718, pp. 244–251. Springer, Heidelberg (1993)
10. Neff, A.: A Verifiable Secret Shuffle and its Application to E-Voting. In: Proceedings of ACM CCS 2001, pp. 116–125. ACM Press, New York (2001)
11. Okamoto, T.: An Electronic Voting Scheme. In: Proceedings of IFIP 1996, Advanced IT Tools (1996)
12. Ohkubo, M., Miura, F., Abe, M., Fujioka, A., Okamoto, T.: An Improvement on a Practical Secret Voting Scheme. In: Zheng, Y., Mambo, M. (eds.) ISW 1999. LNCS, vol. 1729, pp. 225–234. Springer, Heidelberg (1999)
13. Chaum, D.: Elections with Unconditionally-Secret Ballots and Disruption Equivalent to Breaking RSA. In: Günther, C.G. (ed.) EUROCRYPT 1988. LNCS, vol. 330, pp. 177–182. Springer, Heidelberg (1988)
14. Pfitzmann, B., Waidner, M.: Unconditionally Untraceable and Fault-tolerant Broadcast and Secret Ballot Election. Hildesheimer informatikberichte, Institut fur Informatik (1992)
15. Dolev, D., Strong, H.: Authenticated Algorithems for Byzantine Agreement. SIAM Journal on Computing 12(4), 656–666 (1983)

16. Cachin, C., Kursawe, K., Shoup, V.: Random oracles in Constantinople: Practical Asynchronous Byzantine Agreement using Cryptography. Journal of Cryptology 18(3), 219–246 (2005)
17. Pedersen, T.P.: Non-interactive and Information-Theoretic Secure Verifiable Secret Sharing. In: Feigenbaum, J. (ed.) CRYPTO 1991. LNCS, vol. 576, pp. 129–140. Springer, Heidelberg (1992)
18. Cramer, R., Damgård, I., Schoenmakers, B.: Proof of Partial Knowledge and Simplified Design of Witness Hiding Protocols. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 174–187. Springer, Heidelberg (1994)
19. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
20. Bellare, M., Rogaway, P.: Random Oracles are Practical: a Paradigm for Designing Efficient Protocols. In: Proceedings of ACM CCS 1993, pp. 62–73. ACM Press, New York (1993)
21. Bellare, M., Garay, J.A., Rabin, T.: Fast Batch Verification for Modular Exponentiation and Digital Signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 236–250. Springer, Heidelberg (1998)

Resilience Strategies for Networked Malware Detection and Remediation

Yue Yu[1], Michael Fry[1], Bernhard Plattner[2],
Paul Smith[3], and Alberto Schaeffer-Filho[4]

[1] School of Information Technologies, University of Sydney, Australia
{tinayu,mike}@it.usyd.edu.au
[2] Computer Engineering and Networks Laboratory, ETH Zurich, Switzerland
plattner@tik.ee.ethz.ch
[3] Safety and Security Department, AIT Austrian Institute of Technology
paul.smith@ait.ac.at
[4] School of Computing and Communications, Lancaster University, UK
asf@comp.lancs.ac.uk

Abstract. Network propagated malware such as worms are a potentially serious threat, since they can infect and damage a large number of vulnerable hosts at timescales in which human reaction is unlikely to be effective. Research on worm detection has produced many approaches to identifying them. A common approach is to identify a worm's signature. However, as worms continue to evolve, this method is incapable of detecting and mitigating new worms in real time. In this paper, we propose a novel resilience strategy for the detection and remediation of networked malware based on progressive, multi-stage deployment of resilience mechanisms. Our strategy monitors various traffic features to detect the early onset of an attack, and then applies further mechanisms to progressively identify the attack and apply remediation to protect the network. Our strategy can be adapted to detect known attacks such as worms, and also to provide some level of remediation for new, unknown attacks. Advantages of our approach are demonstrated via simulation of various types of worm attack on an Autonomous System infrastructure. Our strategy is flexible and adaptable, and we show how it can be extended to identify and remediate network challenges other than worms.

Keywords: network resilience, worm detection, worm remediation.

1 Introduction

Worms represent a large class of networked malware. Worms are self-replicating, self-propagating malware that can pose a serious threat to networks. Some worms can spread at great speed, infecting and potentially damaging a large number of hosts in a very short time, so that human reaction is unlikely to be effective. There is a need to develop new mechanisms capable of detecting and reacting to network propagated attacks in real time. Research on worm detection has produced many approaches to identify them. A common approach is to identify a worm's signature. However, in some network environments the achievement of real-time signature detection can be hampered by a lack of computational

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 233–247, 2012.

resources. Furthermore, as worms continue to evolve, this method is incapable of detecting and mitigating new worms in real time.

We characterise worm attacks, particularly in their propagation phase, as a type of network *challenge*. Maintaining the normal operation of networks, faces many types of challenges, including other malicious attacks such as DDoS, equipment failures, operational overloads and mis-configurations. Our broad aim is to make networks more *resilient* to such challenges. Resilience is the ability of the network to maintain acceptable levels of operation in the face of challenges. Our work on resilience is considered in the context of a general two-phase high-level strategy, called $D^2R^2 + DR$: *Defend*, *Detect*, *Remediate*, *Recover* + *Diagnose*, *Refine* [1]. Central to the strategy is the real-time management and reconfiguration of interacting detection and remediation mechanisms.

There are a number of mechanisms that can be used to meet our aim of network resilience. Detection mechanisms, such as link monitors and anomaly detectors, assist in the identification and categorisation of challenges such as worms. Remediation mechanisms, such as rate limiters and firewalls, are used in the subsequent mitigation of these challenges. Recently, we have proposed a *policy-based, multi-stage resilience approach* [9], in which the configuration of detection and remediation mechanisms deployed in the network is dynamically refined as new information about challenges becomes available.

In this paper, we use this approach to manage a range of resilience mechanisms to combat networked malware attacks such as worms. We use policies to control the operation of such mechanisms, and how they should be reconfigured in the face of different attack behaviours. Instead of relying on known payload attack signatures, which is the most widely deployed worm detection method, we show how our approach can embrace and adapt a range of detection and remediation mechanisms for both known and unknown attacks. In our case studies, changes in the distribution of specific traffic features are monitored, and a set of active policies determines how this information should be interpreted to contain worm propagation. Resilience strategies are evaluated using a *policy-driven simulation environment* [8]. The primary contribution of this paper is a demonstration of the generality and benefits of our approach for dealing with an ever changing class of network challenge in the form of worms. We also show how our approach may easilly be extended to deal with other forms of networked malware such as port scans.

The remainder of this paper is organised as follows: Section 2 presents related work on worm attack management. Section 3 provides an overview of background work on policy-based resilience management. Section 4 outlines our multi-stage, policy-driven approach applied to the detection and remediation of worms. Section 5 presents results from the simulated deployment of worm resilience strategies for a number of known worm attacks, and also shows how the flexibility of our framework assists the evolution of resilience strategies to meet new challenges. Finally, Section 7 presents concluding remarks.

2 Related Work

Worms are still a widespread and persistent form of malware, attacking targets via various channels and strategies. Historically, most worms have propagated by scanning random IP addresses to infect vulnerable hosts. The worm will send an infectious payload to the target port of hosts running the desired services. In 2001, the code red worm infected 360,000 hosts in 10 hours [2]. In 2003, the "SQL Sapphire Slammer" [17] worm infected tens of thousands hosts in less than half an hour. In 2004, Cabir appeared as the earliest known worm targeting mobile phones. Now, worms infect smart phones via the Internet, Storage Cards, SMS, MMS, and even Bluetooth. Spoofed SMS messages are used to steal personal bank information by the ZeuS MitMo [11]. In May 2012, trend micro reported that the "WORM_STEKCT.EVL" worm spreads via facebook and instant messengers [12]. Also in May 2012, Iranian government computers were attacked by Flame, which is complex modular malware that includes worm-like features. It is said to have a "close relation" with the previous Stuxnet and Duqu worms [13]. Port scans and worms are still a major attack vector for infection and propagation by botnets. Thus worm attack detection and remediation remain serious issues, as is the ability to adapt strategies in the face of evolving malware.

There have been a number of surveys of worm detection [3]. The focus is generally on detecting worm attacks during the propagation phase. The most common approach is based on signatures, which monitor the payload of packets to detect sequences of known worms. Examples of signature-based detection include *snort* and *bro*, sonicwall IPS, and checkpoint IPS. An alternative approach is to monitor protocol sequences of known worms, such as unsuccessful TCP connection attempts. However, signature-based detection suffers from scaling problems. As the number of different worm types grows, so too do the signature databases that are referenced by detection systems, increasing the overhead of real-time detection. Furthermore, in certain resource-constrained environments, such as wireless mesh networks, conventional signature detection systems are not feasible due to their overheads [6].

Signature-based detection is limited to detecting known worms. It cannot detect unknown attacks. Anomaly-based detection observes network traffic looking for abnormal behaviour and generates alarms for anomalies. The challenges lie in distinguishing normal and anomalous behaviour, and the setting of thresholds to detect anomalies. If not designed carefully, such systems could generate many false alarms [3]. There are also trade-offs between accuracy and complexity. Simple tools such as volume monitors will detect large-scale attacks, but may also generate alarms for normal traffic. Alternatively, more sophisticated and accurate detectors may incur unreasonable overheads when operating in real time.

Previous work has also addressed worm containment. In [4] an approach to slowing worm propagation is proposed. However, it would be desirable to achieve complete elimination once a particular attack is identified. Another method of containment is address blocking. Once a host is identified as a scanner, all traffic from this host will be throttled [5]. This approach does not scale well to large-scale attacks from multiple sources.

In the work reported here we overcome the limitations of existing techniques by progressively monitoring a range of traffic features. Worm type anomalies can be identified in real time at an early stage using incomplete information. Interim remediation strategies can be applied until full or partial identification of the worm type is concluded. Furthermore, we offer a platform that enables ongoing development and refinement of malware detection and remediation strategies.

3 Background: Policy-Based Resilience Management

We rely on a policy-based approach to monitor and react to various network challenges. Through policies we can decouple the *"hard-wired"* implementation of resilience mechanisms used to combat a specific attack from the run-time management system that defines their role in a resilience strategy. Consequently, resilience strategies can be adapted without interrupting service operation.

When defining a resilience strategy, there are trade-offs between the overheads, timescales and accuracy of available mechanisms for challenge detection. Our assumption is that detection that yields coarse-grained findings, e.g. detect the presence of an anomaly, are more timely and have a lower overhead. Fine-grained information, e.g. details of the nature of the anomaly, allows better decisions regarding how to mitigate a challenge, but this information is derived using mechanisms with a higher overhead. Consequently, we advocate and utilise a multi-stage resilience approach [9], which is based on the successive activation of mechanisms to analyse and remediate a challenge, from initial detection through to eventual identification. This approach is illustrated in Fig. 1.

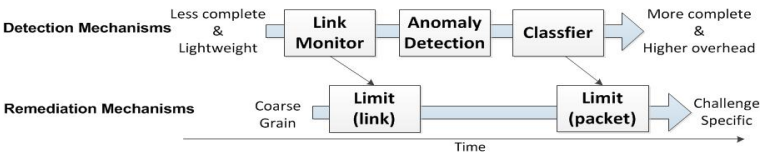


Fig. 1. Coarse to fine grained challenge identification and remediation

Initial detection is triggered by lightweight mechanisms, such as a link monitor. The detection is coarse grain, by this we mean it does not identify detailed information about an attack, such as source and destination addresses. Then more sophisticated mechanisms are invoked. This approach allows the resilience mechanisms to adapt to the prevailing resources, timeliness and accuracy. This is necessary, as the simultaneous operation of a large number of challenge-specific detection techniques can be too resource intensive. Through offline and/or online challenge analysis it is possible to collect network metrics and traffic information. For example, there is a significant amount of published information available that can assist understanding of (known) worms. For unknown challenges, online information needs to be gathered and analysed. Ultimately, for each specific challenge, it is then possible to encode a complete resilience strategy into

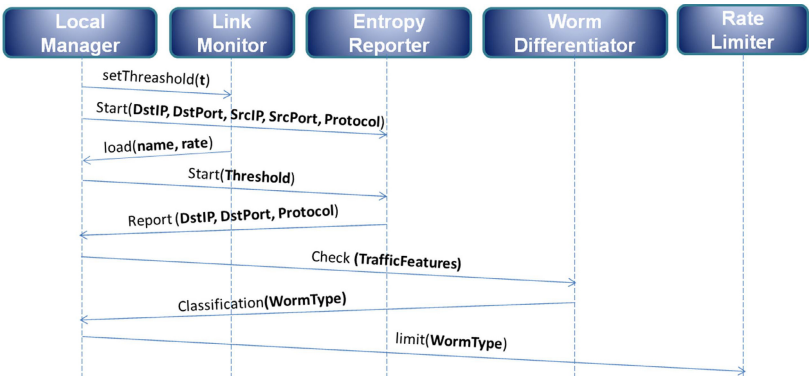


Fig. 2. Algorithm for incremental worm identification and remediation

resilience patterns [7], representing the policy-driven configurations of a set of mechanisms for combating that challenge.

4 A Strategy for Worm Attack Resilience

In this section, we describe how the policy-based resilience management approach can be applied to worm challenges. In our demonstration scenario the goal is to make an ISP or enterprise network - an Autonomous System (AS) - resilient to external worm attacks. It is a border protection strategy, requiring activation of resilience mechanisms at the ingress links to the AS.

4.1 Strategy Overview

While various mechanisms may be used to identify worm-related anomalies, we illustrate our approach using volume-based monitoring and techniques based on *information theory*. In particular, we quantify information using *entropy* [19], which represents the uncertainty associated with the values of the different network traffic features observed. The entropy estimation for anomaly detection relies on the assumption that anomalies will disturb the distribution of certain traffic features in specific ways [20]. The strategy for incremental challenge identification and remediation is outlined in Fig. 2. Resilience mechanisms are realised as a number of policy-enabled *Managed Objects (MOs)* that co-operatively enforce the resilience strategy. A physical device, e.g., a router, can be configured to implement several *MO*s. For example, the *rate limiting* and *entropy reporting* management functions could be colocated in the same physical equipment.

4.2 Policy-Driven Resilience Mechanisms

In the following we detail the operation of each of the resilience mechanisms, implemented as Managed Objects, that collectively realise our strategy.

Local Manager: the *LocalManagerMO* configures the other MOs to achieve the resilience strategy. These are *event-condition-action* (ECA) policies which specify activations and reconfigurations of the MOs. The algorithm for incremental worm identification and remediation is presented in Fig. 2. At start-up the *LocalManagerMO* invokes two MOs. On each of the ingress links to the AS a *LinkMonitorMO* is activated to start monitoring link utilisation, along with a threshold parameter. An *EntropyReporterMO* is also activated along with a list of features that it is to monitor.

Link Monitor: used for evaluating the link utilisation at a given periodicity, with its threshold being set by *LocalManagerMO*. *LinkMonitorMO* notifies *LocalManagerMO* of any sustained traffic above threshold. This aims to rapidly detect fast propagating worms that can generate high volumes of traffic (and can also be used for other high-volumes attacks such as DDoS). The appropriate threshold is established through experimentation, which explores the trade-offs between early detection and accuracy [14]. Slower propagating worms may escape detection by volume, which therefore require another form of early detection.

Entropy Reporter: the *EntropyReporterMO* continuously monitors the dispersion of traffic features using the computationally efficient Shannon's entropy algorithm [18]. The features monitored are source IP, source port, destination IP, destination port and protocol. *EntropyReporterMO* recomputes entropy for the five features every 10 seconds and stores them in a vector. On notification of a volume event, *LocalManagerMO* sets threshold values to *EntropyReporterMO*, which then compares the entropy history of each feature to see if a threshold change has been exceeded. When worms perturb the entropy values of several traffic features beyond the threshold, *EntropyReporterMO* generates an event.

Worm Differentiator: on notification from *EntropyReporterMO* the *LocalManagerMO* activates the *WormDifferentiatorMO* along with details of the perturbed traffic features. The suspicious features are matched against a database of symptoms of known worms. These represent relatively short signatures of worms, compared to the payload-based signatures of existing detection systems. The signature is summarized based on the traffic features captured by EntropyReporter. This reduces the costs of signature matching. If a match is found, an event is notified that identifies the type of worm along with a flow specification for the worm packets. However, if no match is found, for example if it is a new type of worm, then alternate action needs to be taken (see discussion below).

Rate Limiter: on receiving notification from the *WormDifferentiatorMO*, the *LocalManagerMO* invokes the *RateLimiterMO*. This module can shape traffic according to parameters that specify the amount and types of packets to be discarded. It can limit a percentage of all packets on a link. Alternatively, it can limit at the flow level. In this case *RateLimiterMO* discards all packets that conform to the worm's characteristics, thus throttling all attack packets, without having to identify attacking sources.

4.3 Discussion

The policy-based strategy for worm resilience has advantages over traditional solutions. Firstly, through the use of an anomaly-based detection scheme we can potentially detect unknown (zero-day) worms, as will be demonstrated in Section 5.3. Secondly, through the use of policies we can quickly adapt how the system responds to attacks, since the policies are decoupled from the implementation of the managed objects and can be easily changed. Finally, the multi-stage approach introduces intermediate steps, specified by policies, when dealing with attacks. Thus, temporary forms of remediation may be put in place while the challenge is being processed, and until the root cause is reliably identified.

5 Experimentation

To evaluate the performance of resilience strategies we have developed a *policy-driven resilience simulator* [7]. The toolset supports the simulation of a range of network challenges, such as DDoS attacks and worm propagations, and the implementation of our policy-driven approach to combat challenges. Resilience strategies for a particular challenge might use different combinations of mechanisms and policies, and the toolset enables offline evaluation of strategies.

The simulation platform is an integration between a network simulator based on OMNeT++ and *ReaSE* [15] [9], and the Ponder2 policy framework [16]. It allows the specification of policies to define which mechanisms must be activated according to events observed. These mechanisms can be instantiated at any component of the topology. Each instrumented object defines a management interface specifying which operations it supports. Management interfaces are used by Ponder2 for the invocation of operations on the objects. Communication between Ponder2 and OMNeT++ is implemented using XMLRPC. In the remainder of this section, we evaluate a range of detection and remediation strategies using this toolset.

5.1 Worm Challenge Simulation: Identifying and Remediating Known Worm Attacks

We use *ReaSE* to create topologies and generate synthetic traffic loads. The authors of this package have previously demonstrated that the simulated background traffic and the attack traffic are a realistic approximation of traffic observed in real networks [10]. *ReaSE* can generate Code Red Worm propagation [9]. We have extended the package to simulate the Witty and Slammer/Saphire worms, and a port scan.

In order to model a network that is large in scale, we simulated a network consisting of 35 Autonomous Systems (ASes): 26 stub ASes connected by 9 transit ASes. 1616 hosts, 71 web servers and 21 interactive servers generate background traffic. Hosts across the network can be nominated to be *zombies*, which generate worm probing packets. Our strategy is simulated at a stub AS. The various resilience mechanisms are activated on a gateway router of the AS and its ingress link from a core router.

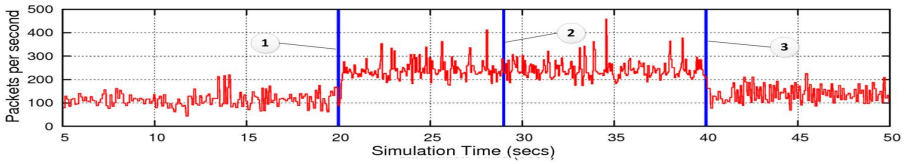


Fig. 3. Simulation results for the Worm resilience strategy

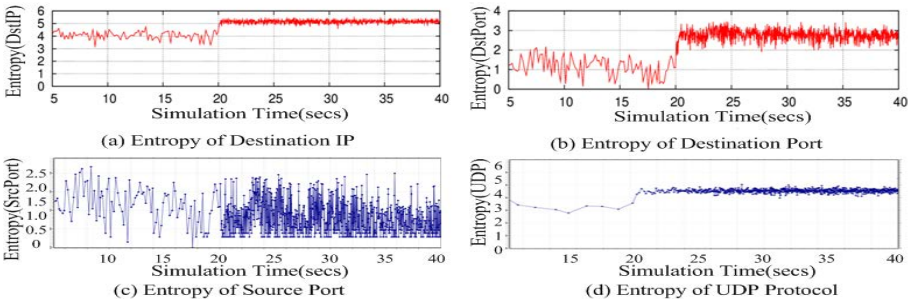


Fig. 4. Entropy changes with Witty worm

Fig. 3 shows the volume of traffic on the ingress link for a simulated, blind-scanning Witty worm attack. In this scenario, the attackers sent a maximum of 8000 probing packets. At the start, LinkMonitor is activated with an alarm threshold set to an increase in average traffic on the link of twice the previous average, which is recomputed every 2500 packets. EntropyReporter is also activated to collect periodically (every ten seconds) packet-level entropy values on five traffic features: destination IP, destination port, source IP, source port and protocol. Fig. 3 shows the worm propagation starting at about 20s(1). Due to the high volume of traffic on the ingress link, an alarm is raised firstly by LinkMonitor at 29s(2). (This form of early detection is appropriate for hig-volume attacks, but a different strategy would need to be deployed for low-volume attacks.) Following this, EntropyReporter is interrogated for any significant changes in the five traffic features. The most recent entropy trend for each feature is computed and compared with the previous average entropy, with results showing that destination IP has become dispersed, destination port dispersed, source port centralised, and UDP protocol more dispersed. These results are reported back for further analysis (3). The WormDifferentiator is then invoked and and can identify these changes in entropy as being a signature of the Witty worm, as descibed below. Consequently, at 40s (3) RateLimiter is configured to filter all probing packets, specified as all UDP packets with a source port 4000.

The features of the Witty worm are that it scans random IP addresses and destination ports using UDP, with a constant source port of 4000. Fig. 4 shows the changes in entropy of four of the features following the onset of the attack at 20s. They indicate characteristics of this specific worm attack. Furthermore, the

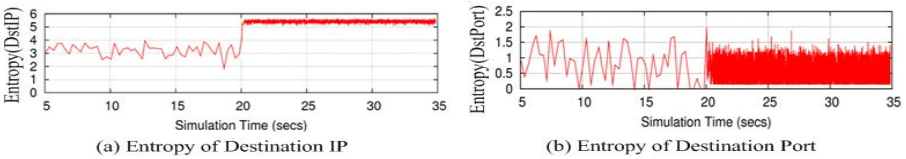


Fig. 5. Entropy changes with Code Red worm

separation between resilience policies and mechanisms permits easy adaptation of these parameters in an operational context, should conditions change.

In contrast we also show partial results of a simulated Code Red attack. The scenario is identical to the Witty simulation above. Code Red also scans random IP addresses, but always to destination port 80. These features are indicated in Fig. 5 where, like Witty, the destination IP becomes more dispersed, but in this case the destination port becomes more concentrated.

5.2 Policy-Based Adaptation of Resilience Strategies

Our simulation environment enables us to experiment with parameter settings and test the impact of adaptations of the strategies by modifying policies, which are executed by the LocalManager. As an example, worm attacks can generate high volumes of traffic which lead to denial of service. We may therefore want to consider adapting our strategy by introducing some early, interim remediation.

A simple adaptation would be for the LocalManager, at the first notification from the LinkMonitor at 29s, to also invoke the RateLimiter to shape all incoming packets. This coarse-grained remediation remains until the worm has been identified and attack specific remediation takes over. Fig. 6 shows the Witty attack scenario with such progressive remediation. Between 29s and 40s, 30% of network traffic on the link is blocked initially. At 40s, the WormDifferentiator matches the Witty signature then only the malicious traffic is blocked. Our platform provides results that show about 28% of benign traffic being blocked during the period of interim mitigation and 32% of malicious worm packets. We do not claim these to be ideal results but they are illustrative of our platform. Off-line risk analysis would determine if the costs (lost benign packets) are appropriate to the benefits (reduced flooding). Our platform provides inputs to such analysis, including the applications that are affected due to the blocking.

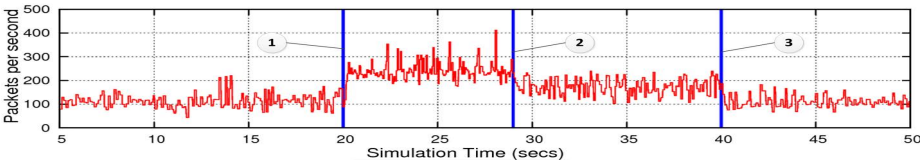


Fig. 6. Simulation results for the Worm resilience strategy with progress mitigation

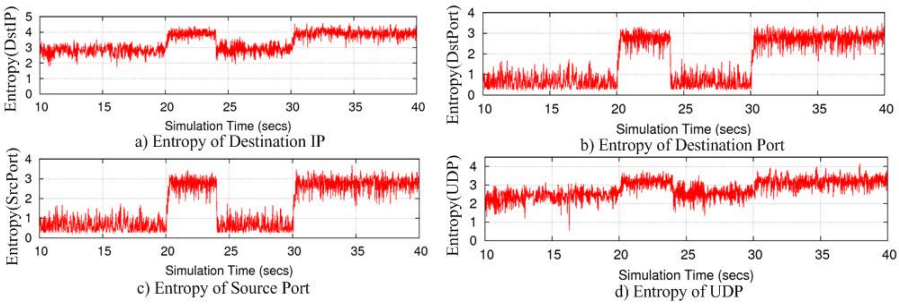


Fig. 7. Entropy changes with New worm

5.3 Resilience against Unknown Attacks

We now demonstrate the generality of our resilience approach to combat unknown attacks. We illustrate this by fabricating a new worm. For example, a new worm might try to escape detection in two ways. Firstly it mutates an exisiting worm to present new traffic distribution features. In our case we mutate the Witty worm to gnerate random source ports in attack packets. Secondly. the worm tries to camouflage itself by hibernating for some time during its propagation. In this case, the worm propagation commences at 20s but lasts for just 4 seconds, then it hibernates before continuing at 30s. It spreads rapidly by sending malicious UDP packets with payload size 600 bytes from random source ports to random destination ports. Fig. 7 illustrates the traffic feature changes. The destination IP becomes more dispersed during propagation. Both source port and Destination port entropy become significantly more dispersed. UDP protocol entropy is more dispersed while TCP protocol is more concentrated.

The new worm features are captured by EntropyReporter but the WormDifferentiator will find no matched signature for them. Then a choice of actions to be taken can be defined by policies. A human operator should be alerted in these cases with a report. Should this indeed be a new worm, then it will be a simple matter for the operator to add the new signature. The new signature is generated based on the traffic features gathered from EntropyReporter. Even without a signature, some automatic remediation can be implemented to protect the network.

For example, we implemented and simulated the following policy-based remediation strategy. For this unknown attack, EntropyReporter infers that the suspected worm is carried in UDP packets but from a large number of suspicious source ports and destination ports. Therefore we further analyse the frequency of suspicious packets by sender host. This could be monitored by the receiving hosts. However this approach will increase computational overheads and is not easy to deploy in a large scale enterprise environment. So a better solution is to embed this strategy in the border router, and activate it once an attack is suspected. Thus, after the EntropyReporter informs that malicious packets are UDP packets, further analysis is performed to monitor the source IP address for all

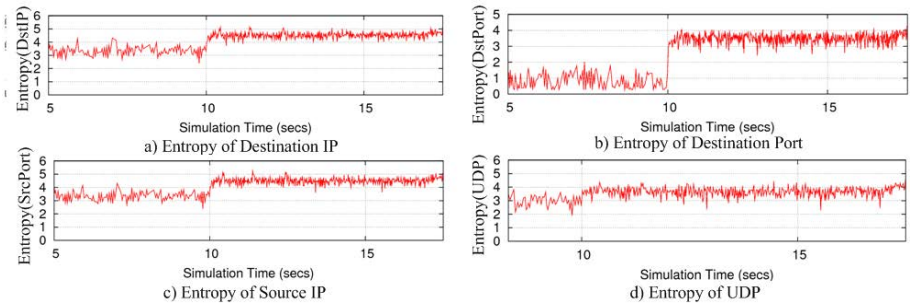


Fig. 8. Entropy changes with Port Scan

incoming UDP packets. Source IP addresses appearing at a significantly higher than average frequency are added to a blacklist and RateLimiter is configured to block all packets from blacklisted sources.

Results for this initial strategy show that, with our background load of benign traffic, it produces 40% false positive rate (percentage of benign traffic blocked) while mitigation eventually achieves zero false negatives. Our platform permits further experiment to refine strategies, including adjustment of alarm and traffic shaping thesholds, and produces results whereby strategies can be evaluated.

5.4 Resilience against Other Malware: Port Scan

We finally show how our worm resilience strategy can be adapted for another class of known network malware. Port scans are a reconnaisance phase of many network attacks. Typically, an attacker searches for potentially vulnerable hosts by trying to connect to ports across random IP addresses. In order to further explore the generality of our approach, we experimented with a port scan scenario under the same topology. In this scenario the attacker scans UDP ports across the well-known port range of 0-1023, attempting to open connections to these ports. Fig. 8 shows the attack starting at 20s, with subsequent changes to traffic features. Destination port and destination IP entropy increase substantially due to scanning across many hosts and ports. UDP entropy increases.

Once a potential attack is identified we can attempt to apply mitigation. In our experiment the strategy deployed is similar to the previous worm remediation. In this case, once an alarm is generated from LinkMonitor, the LocalManager will trigger RateLimiter to initially block 30% of network traffic on the link. We then commence analysis of all incoming UDP packets and, as before successively block packets from high frequency source hosts.

6 Evaluating Our Approach to Network Resilience

We have described a novel solution that enables the progressive multi-stage deployment of resilience mechanisms, based on incomplete challenge and context

information. We have further illustrated the flexibility of our approach when adapting to new challenges in the form of zero-day malware. We have also shown how our approach enables experimentation and development of new resilience strategies. Here we summarily evaluate our approach from different aspects.

6.1 Generality, Efficiency and Scalability

A central problem for the automatic identification and remediation of networked malware is that these challenges display a wide array of features and are ever changing in their characteristics in order to evade detection. Our approach allows detection strategies to be composed such that a range of Managed Objects can be applied to early detection and subsequent identification. For example, malware that propagates at high rates can initially be detected via a link monitor. More stealthy, slower propagating malware can evade volume based detection, but can possibly be detected by monitoring entropy of traffic features. We have performed initial experiments with a simulated stealth worm that generates observable changes in traffic features. However, the development of feasible and accurate strategies for this scenario is subject to future work. In general, we propose that our approach is capable of adopting and applying a wide range of algorithms that have been developed in the literature.

Our multi-stage approach permits deployment of strategies for real-time detection without incurring the overall costs of other approaches. Always-on, real-time monitoring is relatively lightweight. While more complex analysis is only invoked at a point where there is confidence that the network is under challenge. The approach also enables distributed detection, identification and remediation. Under the control of a policy-based local manager, managed objects can be configured and invoked at multiple vantage points in the network. This can be applied at the level of sub-networks or across an entire domain.

6.2 Early Detection vs. Accuracy

Our method is based on packet level analysis, which has the advantage of early detection of potential attacks. Many previous approaches are based on flow analysis. However, routers or switches only export flows after a ramp-up period (typically 15 seconds or more). A further period is then required for anomolay detection algorithms to analyse an attack. In this time, the whole network will be at risk [21]. Our packet based approach can detect large-scale attacks more quickly and can apply interim remediation. But this may be at the expense of accuracy, resulting in high false positive rates.

A benefit of our approach is the ease with which it enables experimentation and refinement of these trade-offs. We can develop and refine strategies through the configuration of different mechanisms via policy and the setting of parameters such as thresholds. Our platform generates results that enable evaluation of strategies. For example, in Table 1 we summarise the features analysed in our scenarios and the results of the remediation strategies deployed. Further experimentation can identify the trade-offs between early detection and accuracy.

Table 1. Performance evaluation with different malware

	Code Red	Witty Worm	Unknown Worm	Port Scan
Entropy(DstIP)	Increase	Increase	Increase	Increase
Entropy(DstPort)	Major Decrease	Increase	Major Increase	Major Increase
Entropy(SrcPort)	Increase	Major Decrease	Major Increase	Decrease
Entropy(UDP)	Decrease	Increase	Increase	Increase
Blocked benign	1%	0	40%	20%
Allowed malicious	0	0	0	0

New mechanisms can be added to analyse additional traffic features, such as volumes at different levels of granularity or protocol sequences.

6.3 Flexibility and Evolution

Policies are deployed to coordinate different managed objects, and can be configured in a flexible and simple way via parameterisation. Our platform has the flexibility to enable adaptation of strategies to fit different detection scenarios and the inclusion of new mechanisms. For example, a Managed Object to analyse packet payload content could be added as a stage in the identification of a stealth attack.

New challenges will continue to emerge in networks. We have previously shown how our platform can be used to develop resilience strategies against volume anomolies such as DDoS attacks [9]. In doing so we easily re-used some of the managed objects described here, such as LinkMonitor and RateLimiter. We envisage that our approach can also be used to experiment with and develop resilience strategies for other forms of challenge, such as network faults. In general, our approach facilitates development and deployment of network resilience strategies, which can be further refined as new challenges emerge and/or our understanding of how to achieve resilience is enhanced.

7 Conclusions and Future Work

This paper has demonstrated our policy-based, multi-stage approach for the detection and remediation of network malware, using worm attacks as a primary example. In our scenarios, this approach identifies known worms on the basis of signatures that are defined as entropy perturbations of certain features. It is also capable of detection of new worms and enables partial remediation of such attacks. We have shown how we can extend the system for other forms of networked malware such as port scans.

Our approach overcomes drawbacks of existing approaches discussed under related work, by operating only lightweight detection mechanisms while a network is operating normally. More heavyweight mechanisms are only invoked once the early symptoms of a potential worm attack are detected. Thus high overhead containment is only used when there is confidence that a network is

under attack. We also enable early remediation of attacks while the precise nature of the attack is diagnosed, followed by attack-specific remediation once the attack is understood. We suggest that our approach is therefore potentially more resource-efficient than existing systems, while at the same time no less accurate.

We have explored the application of our approach to a range of existing worm attacks, and also an unknown type. Thus we have demonstrated significant advantages of flexibility and evolvability. Through the separation of policies and mechanisms we are able to re-use and refine our resilience strategies and mechanisms. New worm types can be accommodated by micro-level adaptations such as traffic feature sets and thresholds. Our system supports the evolution of identification and remediation strategies as network contexts change. Finally, we have shown how our approach supports experimentation through simulation of resilience strategies, mechanisms and parameters prior to actual deployment.

Ongoing work is studying how better to quantify the performance trade-offs of our approach and thus how best to optimise detection and remediation of the many challenges that networks face. We have previously shown the initial application of our approach to a different form of challenge – a DDoS attack [9]. We need to further explore the generality of our approach to more stealthy forms of attack. Future work will also focus on developing resilience strategies that integrate the simultaneous detection, identification and remediation of multiple challenge types including worms, DDoS, flash crowds, network faults, etc. This may introduce issues of policy conflict and resolution, and resource trade-offs.

Acknowledgments. This research is supported by the European Union Research Framework Programme 7 via the ResumeNet project with contract number FP7–224619, NICTA (National ICT Australia). The authors would like to thank David Hutchison, Andreas Mauthe for their contribution to this work.

References

1. Sterbenz, J.P.G., Hutchison, D., Cetinkaya, E.K., Jabbar, A., Rohrer, J.P., Scholler, M., Smith, P.: Resilience and survivability in communication networks: Strategies, principles, and survey of disciplines. Comput. Netw. (2010)
2. Yu, W., et al.: On Defending Peer-to-Peer System-based Active Worm Attacks. In: IEEE Global Telecommunications Conference, pp. 1757–1761. IEEE Press (2006)
3. Li, P., Salour, M., Su, X.: A survey of Internet worm detection and containment. IEEE Communications Surveys & Tutorials 10(1), 20–35 (2008)
4. Chen, C., Chen, Z., Li, Y.: Characterizing and defending against divide-conquer-scanning worms. Computer Networks 54(18), 3210–3222 (2010)
5. Chen, S., Tang, Y.: DAW: A distributed antiworm system. IEEE Transactions on Parallel and Distributed Systems, 893–906 (2007)
6. Hugelshofer, F., Smith, P., Hutchison, D., et al.: OpenLIDS: a lightweight intrusion detection system for wireless mesh networks. In: MobiCom 2009. ACM, USA (2009)
7. Schaeffer-Filho, A., Smith, P., Mauthe, A., Hutchison, D., Yu, Y., Fry, M.: A Framework for the Design and Evaluationof Network Resilience Management. In: 13th IEEE/IFIP Network Operations and Management Symposium, USA (2012)

8. Schaeffer-Filho, A., Smith, P., Mauthe, A.: Policy-driven network simulation: a resilience case study. In: SAC 2011, Taiwan, pp. 492–497 (March 2011)
9. Yu, Y., Fry, M., Schaeffer-Filho, A., Smith, P., Hutchison, D.: An adaptive approach to network resilience: Evolving challenge detection and mitigation. In: DRCN 2011: 8th International Workshop on Design of Reliable Communication Networks, Poland, pp. 172–179 (October 2011)
10. Gamer, T., Scharf, M.: Realistic Simulation Environments for IP-based Networks. In: Proceedings of the OMNeT++ Workshop, Marseille, France (March 2008)
11. La Polla, M., Martinelli, F., Sgandurra, D.: A Survey on Security for Mobile Devices. IEEE Communications Surveys Tutorials (99), 1–26 (2012)
12. Pantanilla, C.: Worm Spreads via Facebook Private Messages, Instant Messengers, Malware blog, Trend Micro (May 2012)
13. Flame worm one of the most complex threats ever discovered. Virus Bulletin Fight Malware and Spam (May 2012)
14. Brauckhoff, D., Salamatian, K., May, M.: A signal processing view on packet sampling and anomaly detection. In: INFOCOM, pp. 713–721. IEEE Press, USA (2010)
15. Varga, A., Hornig, R.: An overview of the OMNeT++ simulation environment. In: SIMUTools 2008 ICST, Belgium, pp. 1–10 (2008)
16. Twidle, K., et al.: Ponder2 - a policy environment for autonomous pervasive systems, pp. 245–246. IEEE Computer Society, USA (2008)
17. Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S., Weaver, N.: Inside the Slammer worm. IEEE Security and Privacy Magazine 1(4), 33–39 (2003)
18. Zesheng, C., et al.: An Information-Theoretic View of Network-Aware Malware Attacks. IEEE Transactions on Information Forensics and Security, 530–541 (2009)
19. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27, 379–423 (1948)
20. Lakhina, A., Crovella, M., Diot, C.: Mining anomalies using traffic feature distributions. SIGCOMM Comput. Commun. Rev. 35(4), 217–228 (2005)
21. Packet vs flow-based anomaly detection. Technical White Paper, ESPHION Neetwork Disaster Protection (2005)

Detecting Spammers via Aggregated Historical Data Set

Eitan Menahem, Rami Pusiz, and Yuval Elovici

Ben-Gurion University,
Telekom Innovation Laboratories
Information System Engineering Department
Be'er Sheva, 84105, Israel
{eitanme,puzis,elovici}@post.bgu.ac.il

Abstract. In this work we propose a new sender reputation mechanism that is based on an aggregated historical dataset, which encodes the behavior of mail transfer agents over exponential growing time windows. The proposed mechanism is targeted mainly at large enterprises and email service providers and can be used for updating both the black and the white lists. We evaluate the proposed mechanism using 9.5M anonymized log entries obtained from the biggest Internet service provider in Europe. Experiments show that proposed method detects more than 94% of the Spam emails that escaped the blacklist (i.e., TPR), while having less than 0.5% false-alarms. Therefore, the effectiveness of the proposed method is much higher than of previously reported reputation mechanisms, which rely on emails logs. In addition, on our data-set the proposed method eliminated the need in automatic content inspection of 4 out of 5 incoming emails, which resulted in dramatic reduction in the filtering computational load.

1 Introduction

The battle between email service providers and senders of mass unsolicited emails (Spam) continues to gain traction. Surveys show that in recent years 76% to 90% of all email traffic can be considered abusive [12]. A major portion of those billions of Spam emails annually are automatically produced by botnets [14]. Bots create and exploit free web-mail accounts or deliver Spam emails directly to victim mailboxes by exploiting the computational power and network bandwidth of their hosts and sometimes even user credentials. Many Spam mitigation methods are used by email service providers and organizations to protect mail boxes of their customers and employees respectively. There are three main approaches for Spam mitigation; content-based filtering (CBF), real-time blacklisting or DNS based blacklists, and sender reputation mechanisms (SRM).

Real-time blacklists (RBL) are IP-based lists that contain IP prefixes of Spamming MTAs and are regarded as network-level filters. Filtering with RBL is very fast since the decision to accept or reject the email does not require receiving the full email (saving network resources) nor processing its content (saving computational resources). In order to avoid misclassification, RBLs must be updated

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 248–262, 2012.

systematically. For example, Spamhaus [20], Spam-Cop [19], and SORBS [18] are some initiatives that keep RBL systems updated by tracking and reporting spammer IPs. RBL methods, however, cannot solve the Spam email problem entirely, as spammers can escape them by, for example, repeatedly changing their IPs by stealing local network IPs , or by temporarily stealing IPs using BGP hijacking attacks [14].

In comparison to RBL, the CBF are considered much more accurate, but they are also more computationally intensive. In order to speed up the email filtering process, organizations maintain blacklists of repeated Spam senders. Those blacklists usually complement existing CBF methods by performing the first rough filtering of incoming emails. Organizations that choose to maintain their own blacklists gain flexibility in blocking certain addresses, the ability to respond to emerging Spam attacks in real-time, and the flexibility to react immediately if they receive complaints about emails not reaching their destination.

Sender Reputation Mechanisms (SRM) are a collection of methods which compute a liability score for email senders. The computation is usually based on information extracted from the network or transport level, social network information, or other useful information sources. According to Alperovitch et. al [1], sender reputation systems should react quickly to changes in sender's behavioral patterns. That is, when sender's sending patterns take a shape of a Spammer, his reputation should decrease. If the reputation of a sender is below a specified threshold, the system should reject his mails, at least until the sender gains up some reputation by changing his sending properties.

The primary objective of the presented work is to automatically learn a machine-learning based SRM for updating the black and the white lists in a timely manner. We assume that there are differences in the behavioral patterns of spammers and legitimate IPs and try to capture these differences. We also assume that the behavioral patterns of IPs may change over time due to adversary actions, such as stealing local network IPs of benign MTAs, by temporarily stealing IPs using BGP hijacking attacks, or by installing a bot agent at a end-user's device. We further assume that an analysis of a long period of time is important for blacklisting repeated spammers. However the recent behavior is important for rapid reaction to changes in the sender behavior. Table 1 presents the prominent previous works in spam mitigation.

Our contribution is two-fold. We propose a novel method for extracting aggregate features by breaking the email sending history into exponentially growing time windows. We later show that these aggregate features incurs a better IP classifiers, compared to the case when the aggregate features are extracted from the whole email sending history. The second contribution is a new sender IP behavior measure, *Erraticness*, which captures the behavior transitions of a sender IP over a historical time frame (from sending *Ham* to *Spam* and vice versa). The *Erraticness* measure was shown to produce a lower false positive rate, in comparison to the traditional *Spammingness* measure.

2 Related Works

In this section we discuss some sender reputation mechanisms that share a similar problem domain as the proposed HDS based method. Several works have used machine learning algorithms to compute sender reputation from data sets which are not based on email content analysis.

Table 1. Spam mitigation techniques

CBF		[9,23]
RBL / DNSBL		[18,19,20]
SRM	Network level	[14,21,15]
	Transport level	[3,13]
	Application level	[2,4,6,10]
	Spatio-temporal	[8,22,17]

Ramachandran et al. [15] present the SpamTracker that classifies email senders based on their sending behavior rather than the MTA's IP addresses. The authors assume that spammers abuse multiple benign MTAs at different times, but their sending patterns tend to remain mostly similar even when shifting from one abused IP to another. SpamTracker extracts network-level features, such as received time, remote IP, targeted domain, whether rejected, and uses spectrum clustering algorithms to cluster email servers. The clustering is performed on the email destination domains. The authors reported a 10.4% TPR when using SpamTracker on a data set of 620 Spam mails which were missed by the organization filter and eventually were blacklisted in the following weeks.

Sender reputation mechanisms are not only limited to network-level features; reputation can also be learned from a much higher communication level, such as the social network level. The social-network-based filtering approach takes advantage of the natural trust system of social networks. For example, if the sender and the receiver belong to the same social group, the communication is probably legitimate. On the other hand, if the sender does not belong to any trustworthy social group, he is more likely to be blacklisted. There are many methods which make a good use of social networks to create both black and white lists. Few examples are J. Balthrop et al. [2], which used email address books to compute sender trust degree, and Boykin and Roychowdhury [4], which reported 56% TPR with the black list and 34% TPR with the white list by extract social network's features from the email header fields such as From, To, and CC.

The downside of using social network level features is that both the black and white lists can be made erroneous. For example, the black list can be fooled by attackers which use spyware to learn the user's frequently used address list and have one or more of them added to the Spam mail so that the co-recipients (the Spam victims) will look like they belong to the user's social network. The white list can also be fooled by sending Spam mail by using one or more of the user's friends accounts. This can be done, for example, if the friend's computer has been compromised by a bot which selectively send Spam mails.

Beverly and Sollins [3] investigated the effectiveness of using transport-level features, i.e., round trip time, FIN count, Jitter, and several more. The best features were selected using the forward fitting method to train a SVM-based

classifier. They reported 90% accuracy on 60 training examples. However, one of the weaknesses of their method, when compared to RBL and other reputation mechanisms, is that emails (both Spam and Ham) must be fully received in order to extract their features, thus making the Spam mitigation process less effective.

SNARE, by Hao et al. [8] presented a method based solely on network-level and geodesic features, such as distance in IP space to other email senders or the geographic distance between sender and receiver. SNARE classification accuracy has been shown to be comparable with existing static IP blacklists on data extracted from McAfee's TrustedSource system [11].

Finally, West et al. [22] introduce a reputation model named PRESTA (Preventive Spatio-Temporal Aggregation) that is used to predict behavior potential spam senders based on email logs. The authors report that PreSTA identifies up to 50% spam emails that have passed the blacklist having 0.66% false positives.

In light of the related work, in this paper we investigate a new method for updating the reputation of sender MTAs. Our method uses several unique aggregate features of email log data, and a new target function, i.e., *Erraticness* to train a model for IPs reputation. We show later, in section 7 that our approach performs better than classifiers that were trained on the same email log, yet without the aggregate features and the "Erraticness" target feature.

3 Sender Reputation Mechanisms Based on Email Logs

In this section we describe two common method for constructing SRM from email-logs. Let IP denote the Internet address of the Sender MTA. We will assume an implicit partition of the IP address into four fields: two fields (MSB and LSB) of in CIDR (Classless Inter-Domain Routing) notation, the subnet identifiers $IP/8$, $IP/16$, $IP/24$, and the host identifier $IP/32$.

Let $EL = \{IP, T, NR, AE, SpamClass\}^m$ be the relational data set representing the email log. It contains one line for each received mail where T is the receiving time, NR is the number of recipients, AE is the number of addressing errors, $Class$ is the binary mail classification (*Spam* or *Ham*) obtained by an oracle, and m is the number of emails. Table 2 is an example of EL. This example will be used to demonstrate the construction of HDS in Section 4.

We will denote the classifier trained from the EL dataset as EL-based SRM. As can be seen from Table 3, the subnet identifiers play significant role in EL-based SRMs. This fact suggests that learning algorithms are able to identify spamming addresses even based on emails that have passed the blacklist of the email service provider.

Let $[T_{start}, T_{end})$ denote a continuous time range starting at T_{start} (inclusive) and ending at T_{end} (exclusive).

Definition 1. *Spammingness of a sender IP (denoted by $Y_{IP,[T_{start},T_{end})}$) is the fraction of Spam emails sent by the IP during the time window $[T_{start}, T_{end})$.*

Let BLT and WLT be the blacklisting and the whitelisting thresholds, respectively, used by the SRM. If the EL-based SRM classifies an EL instance as *Spam*

Table 2. Example of an email log

#	IP	T	AE	Class
1	IP_1	1	6	*Ham*
2	IP_1	1.5	2	*Ham*
3	IP_1	2.8	3	*Spam*
4	IP_1	4.1	0	*Ham*
5	IP_1	5.5	2	*Ham*
6	IP_1	6.3	2	*Ham*
7	IP_1	7.1	57	*Spam*
8	IP_1	7.9	48	*Spam*
10	IP_2	11	2	*Ham*

Table 3. Infogain ranking of EL features

Feature		Rank
IP	/8	0.262
	/16	0.087
	/24	0.056
	/32	0.015
NR		0.344
AE		0.0002
T		0.041

with confidence above BLT the respective IP address is added to the black-list. The white-list is updated symmetrically. A similar approach using a different set of features, was taken in [8].

Another simple SRM which can be applied directly on the EL data set is a heuristic that detects repeated spammers and trusted legitimate email senders. This Heuristic-SRM is based solely on the spammingness of the IP in the past. If the spammingness of an IP in the past was above BLT, the heuristic-SRM blacklists it. Symmetrically, if the spammingness of the IP is below WLT, the IP will be whitelisted. Since most of the IP addresses are either legitimate email senders or potential spammers this heuristic performs quite well in practice if the considered emails sending history is long enough. However, it can hardly detect changes in the sender behavior, therefore cannot react in a timely manner.

Making aggregations over the entire history of an IP may not be optimal as it releases the focus from the most recent statistics that may indicate behavior changes. A better approach is to split the history on an IP into several non-congruent time windows, as described in the next section.

4 Historical Data Set

The idea of HDS in a nutshell is to aggregate email log records across multiple variable length historical time windows in order to capture both short terms and long terms email sending trends. For each historical time window, HDS records contain multiple aggregations of the attributes in the EL data set. The learning algorithm is applied on the aggregated features to predict the behavior of an IP in the near future. Based on this prediction, HDS-based SRM will update the black and white lists. Intuitively, one special property of the proposed method is its ability to model the subject behavior over time (which is more informative than only its current state). This information allows detection of 'Spammer' behavior patterns even with only very few Spam emails sent. This unique property can speed-up the blacklisting process, and hence improve the true positive classification rate (TPR).

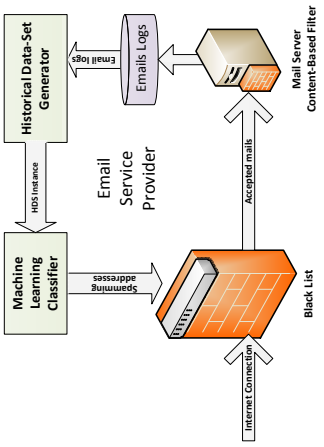


Fig. 1. The HDS work flow

Next, we define the HDS building blocks. Figure 1 depicts the general work flow of HDS based SRM. HDS records are uniquely identified by a reference time T_0 and IP. EL records preceding T_0 are denoted with negative time indexes e.g. T_{-1}. An Historical time window is a continuous range $[T_{-w_0 \cdot 2^i}, T_0)$ where w_0 is the length of the smallest historical time window and i is a positive integer. Using exponentially growing historical time windows gives us two benefits. First, we are able to capture a long history of an IP without exploding the size of the HDS records. Second, the size of the most recent time windows is small enough to detect the slightest change in the behavior of an IP.

Let $FS_{IP,T_0,i}$ be a set of aggregated features of a particular IP, computed using EL records in a historical time window $[T_{w_0 \cdot 2^i}, T_0)$. Each HDS record contains n feature sets $(FS_{IP,T_0,0}, FS_{IP,T_0,1}, \ldots, FS_{IP,T_0,n-1})$. Every feature set $FS_{IP,T_0,i}$ includes aggregates of all features extracted from the email logs. The actual set of features depends on the email service provider and the nature of the email logs. In this paper we have constructed feature sets for the HDS records by taking the sum, mean, and variance of the number of recipients (NR), the number of addressing errors (AE), the CBF processing time (PT), and the $Class$. Note that the mean of $Class$ is in fact the spammingness of the IP in the respective time window. In addition, for each time window we have also included the total number of emails sent and the number of times the sender changed their behavior from sending Spam to sending legitimate emails, and vice versa.

Table 4. The historical dataset (HDS) structure

IP	T_0	Feature Sets			Emerging Behavior
IP_1	1	$FS_{IP_1,1,1}$	$\cdots$	$FS_{IP_1,1,n}$	$Class_{IP_1,1}$
IP_2	2	$FS_{IP_2,2,1}$	$\cdots$	$FS_{IP_2,2,n}$	$Class_{IP_2,2}$
			$\vdots$		

Table 5. Example of a Spammingness-based HDS, derived from data in Table 2

IP	T_0	$[T_{-4}, T_0)$		$[T_{-2}, T_0)$		$[T_{-1}, T_0)$		$[T_0, T_{+4})$
		EC	AE	EC	AE	EC	AE	Spammingness
IP_1	2	-	-	2	8	1	2	0.333
IP_1	4	3	11	1	3	0	0	0.200
IP_1	6	3	5	2	2	1	2	0.400
IP_1	8	5	109	3	107	2	105	0.500

Let $[T_0, T_{Pred})$ be the prediction time window where $Pred$ is its length. Let $Class_{IP,[T_0,T_{Pred})}$ be the target attribute of the HDS records used to train machine learning classifiers. HDS records are identified by an IP and a reference time T_0. They contain n feature sets that correspond to n historical time

windows and a target attribute. Let HDS be a relational data set derived from EL:

$$HDS = (IP, T_0, FS_{IP,T_0,1}, \ldots, FS_{IP,T_0,n}, Class_{IP,[T_0,T_{Pred})})^l$$

where l is the number of records.

Table 4 depicts the historical data set structure. Each HDS record is identified by IP and a reference time T_0 and contains n feature sets.

5 HDS-Based Sender Reputation Mechanisms

Next, we describe two variants of an HDS-based SRM. The target attribute of the first variant is the future *Spammingness* of an IP and the target attribute of the second variant is its future sending behavior *Erraticness*. We will denote these two variants as HDS-based (Spammingness) SRM and HDS-based (Erraticness) SRM, respectively.

5.1 HDS-Based (Spammingness) SRM

In order to train the HDS-based (Spammingness) classifier, we set the target attribute of every HDS record to be the Spammingness of the IP in a time period following T_0 ($Y_{IP,[T_0,T_{Pred})}$). Table 5 shows an example of an historical data set derived from the email log in Table 2.

If the predicted Spammingness is higher than a given threshold, we then apply the BLT threshold on Spammingness (i.e $SpamClass_mean$) in the largest historical time window. Symmetrically, if the predicted Spammingness is lower than the given threshold the WLT, is applied to determine whether or not the IP is added to the white-list.

The operation of HDS-based (Spammingness) SRM resembles the Heuristic SRM described in Section 3. Here, however, the heuristic rule is augmented by the prediction of a machine learning classifier. In Section 7 we will show that this combination produces a high quality SRM. Preliminary experiments (not presented in this paper) show that using the predicted Spammingness only, without applying the BLT and WLT thresholds on historical Spammingness, results in poor classification.

5.2 HDS-Based (Erraticness) SRM

Our second variant of the HDS-based SRM predicts the tendency of an IP to keep his past behavior also during the prediction time window. Next, we define the Erraticness measure, which is used as the HDS target feature.

Definition 2. *Erraticness of a sender IP (denoted by $Z_{IP,[T_{start},T_{end})}$) is the number of behavior changes of an IP during the time window $[T_{start}, T_{end})$.*

We consider a behavioral change as a transition from sending Spam to sending legitimate emails or vice versa. Note that the underlying notion of *Erraticness*,

that MTAs sending behavior can alternate, i.e., MTA can send ham emails for a while, then send some Spam emails, and later turn back to send ham emails, is indeed realistic. In fact, our EL-dataset (discussed in section 6.1) shows that 25.6% of the MTAs changed their sending behavior during a single day (8,804 IPs change their behavior more than 4 times). In section 7 we show that the HDS (*Erraticness*) can learn to avoid blacklisting such MTAs to avoid false-positives. ?

A behavior Erraticness classifier is used as follows: if the classifier's prediction is an unvarying behavior ($Erraticness << \epsilon$), meaning that the IP is not expected to change its behavior in the nearest future, then the IP is blacklisted only if the Spammingness of the IP in the longest historical time window is above the blacklisting threshold BLT. Symmetrically, if the IP Spammingness is below the whitelisting threshold, then the IP is added to the white-list. On the other hand, if the classifier prediction is a behavioral transition, than the IP is blacklisted.

6 Evaluation Setup

It is not possible to directly compare classifiers trained on the EL and the HDS datasets. The main difficulty is that the number of instances and the target-variable in both data sets is different. While each EL instance corresponds to a single email, each HDS instance represents email aggregates for a time period. Therefore, we implemented a unique evaluation environment[1] around WEKA machine-learning tool [7], which enables evaluating the afore mentioned SRMs on a common ground. The evaluation environment simulates the general filtering process of incoming emails at a large ISP.

The blacklisting threshold BLT and the whitelisting threshold WLT, are parameters of the evaluation environment. Their values were empirically chosen to be 0.5 and 0.05 respectively. These values assure low false positive rates while resulting in relatively high true positive rates. Both the black and the white lists were empty in the beginning of all experiments.

6.1 Dataset

To evaluate the proposed SRMs, we made use of a single email log datasets which contained 9.507 million anonymized log entries (emails headers) of 678,509 distincs IPs, which were received during a 168 hours (7 days) period at T-Online ISP. The dataset is comprised of 9 attribues and 12.25% 'Spam' labeled instances. The un-received emails, which were blocked by the T-Online black-list, were not logged and therefore their headers are not included in the dataset. The dataset was fully labeled by an automatic content-based filtering device, eXpurgate [5]. The eXpurgate claims "A Spam recognition rate of over 99%" and "zero false positives" with unpublished false negative rate. The dataset was partitioned into

[1] The code, dataset, and operation instructions can be downloaded from `http://www.iamresearcher.com/data/user/eitan.menahem/`

training and validation sets, each containing the instances of all the emails sent by $200k$ randomly selected sender IPs. The training set (2,835,214 instances) and validation sets (2,864,208 instances) are mutually exclusive, meaning that IPs that exist in the training set do not appear in the validation set, and vice versa.

6.2 Performance Metrics

We use the following performance metrics: %error, true and false positive rates, area under the ROC (Receiver Operating Characteristic) curve, blacklist size, and number of whitelist hits. These performance metrics provide enough information to assess the ability of the HDS-based SRM to both reduce the load from mail servers, and reduce the number of potential customer complaints.

%error is the rate of incorrect predictions and is computed as follows: $\%error = \frac{100*(FP+FN)}{TP+TN+FP+FN}$ where TP, TN, FP and FN stand for the number of true positive, true negative, false positive, and false negative rejections of emails respectively.

The ROC curve is a graph produced by plotting the true positive rate ($TPR = TP/(TP + FN)$) versus the false positive rate ($FPR = FP/(FP + TN)$). The AUC value of the best possible classifier will be equal to unity. The worst possible binary classifier (obtained by flipping a coin for example) has an AUC of 0.5. The AUC is considered as an objective performance metric as it does not depend on the specific discrimination threshold used by a classifier.

Black list size is the number of IPs added to the black list during each experiment execution. The blacklist size mainly affects the IP lookup time and the amount of computational resources spent on its maintenance [16]. Faster lookup times mean less delay in email delivery, while the computational resources required to maintain the blacklist directly translate into cost.

The number of *whitelist hits* is an indication of the number of emails that were delivered without content inspection. A Higher number of whitelist hits means less computational resources spent on Spam filtering.

7 Experimental Results

In the following experiment we study the IP classification performance of the proposed HDS-based SRMs. In the first experiment the SRMs are tested and compared across different machine-learning algorithms. The next experiment study a more realistic scenario, where the black and white lists are updated only once per a time interval. Lastly, in the third experiment we study the effect of the HDS's exponentially growing window length affects on the SRM classification performance.

7.1 The Value of HDS Aggregations

The first question in this study is whether the HDS-based SRM outperforms other related SRMs. To answer this we compare HDS-based SRM to EL-based

SRM on an identical test bed. We argue that in general HDS-based features are at least as informative as EL-based features because they are derived directly from EL-based features and thus HDS features should produce a superior classification model, with respect to that produced using the corresponding EL dataset. In this experiment we simulated a condition in which every email, whose sender's IP is not included in either the black or white lists, is classified by the tested SRMs on arrival. We denote this classification mode as 'continuous mode'.

In order to increase the reliability of the experimental results, we evaluated the HDS-based and EL-Based sender reputation mechanisms on four different machine-learning algorithms from separated machine-learning families. The algorithms used were: Naïve Bayes (Bayes) , C4.5 (Decision Trees) , Logistic Regression (Function) , and BayesNet (Bayes). We also applied the Heuristic SRM as a baseline to compare with other techniques. The HDS instances were generated using the following parameters: $w_0 = 60$ minutes, $n = 5$, and $T_{Pred} = 60$ minutes, i.e., The total history length equals $w_0 \cdot 2^{(n-1)} = 960$ *minutes* (16 *hours*). The time window used by the Heuristic SRM for computing the spammingness of the IP addresses was also set to 960 minutes. The results of this experiment are presented in Table 6.

Table 6. Dependability on ML classifiers

SRM	Learning Algorithm	%Error	TPR	FPR	BlackList		AUC
					Size	Hits	
Hueristic	n/a	3.04	0.761	0.002	5,046	22,379	0.880
HDS-Based (Spammingness)	BayesNet	1.18	0.942	0.006	5,917	31,990	**0.968**
	C4.5	1.85	0.898	0.007	5,577	32,149	0.945
	Logistic	1.20	0.941	0.006	5,913	31,943	**0.968**
	Naïve Bayes	1.46	0.942	0.009	5,918	33,861	**0.967**
HDS-Based (Erraticness)	BayesNet	1.18	0.940	0.005	5,916	31,689	**0.967**
	J48	1.19	0.939	0.005	5,906	31,645	**0.967**
	Logistic	1.25	0.939	0.006	5,910	32,137	**0.967**
	Naïve Bayes	1.44	0.940	0.008	5,916	33,471	0.966
EL	BayesNet	18.99	0.928	0.206	6,565	145,700	0.861
	C4.5	9.92	0.898	0.099	6,116	83,297	0.900
	Logistic	11.04	0.153	0.011	923	11,252	0.571
	Naïve Bayes	10.19	0.488	0.046	3,239	41,233	0.721

Judging the best results for each of the SRMs using Table 6, we can see the HDS-SRM (Erraticness) had the best IP classification performance. Not far behind at second place is the HDS-SRM (Spammingness). Third place, with noticeable AUC difference, is the EL-based SRM. The worst performance was achieved by the Heuristic-based SRM. Interestingly, the best results for each SRM were obtained by different learning algorithms. The Logistic-Regression, which worked very well for both HDS-SRMs, produced the worst results when applied to EL-Based. In this cases, very few IPs were blacklisted by the EL-based SRM, consequently, obtaining a very poor AUC score.

The highest TPR was achieved by the HDS-SRM (Spammingness), whereas the lowest false-positive and error rates were obtained by the HDS-SRM (Erraticness). The EL-based SRM blacklisted the most IPs. However, many of these IPs were of benign senders, which is reflected in the very high false-positive rate obtained by this SRM. We noticed that in some configurations the very simple Heuristic and the EL-based SRMs achieved comparable performance.

7.2 Batch IP Classification

In the previous experiment we exercised a continuous classification mode where the sender reputation is computed and updated each time a new email is received. This mode of operation may not be realistic due to relatively high resource consumption of machine learning based classifiers compared to black and white lists data structures. Moreover, classifying every incoming email would also mean placing the classifier in the critical path and turn it into a bottleneck during the process of handling incoming emails. Another drawback of the continuous classification mode is that most black-lists are optimized for fast information retrieval but do not tolerate frequent updates. Updating the black-list data structure may be a very expensive operation in terms of computational resources [16].

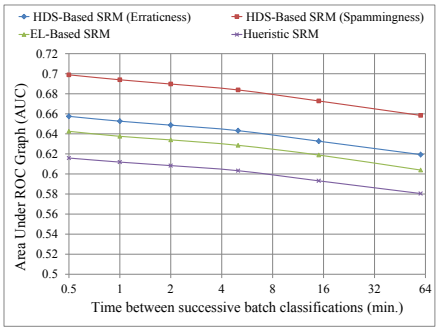


Fig. 2. AUC as a function of time between consequent address-lists updates

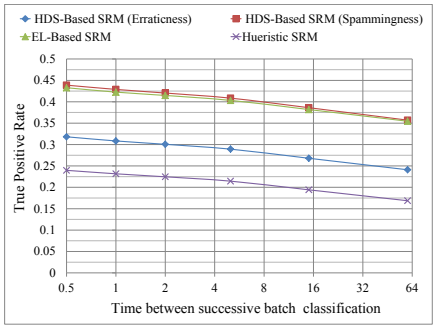


Fig. 3. TPR as a function of time between consequent address-lists updates

It is therefore a good practice to minimize the number of updates and to make them as infrequent as possible. In practice, SRM can be activated once in a while in order to save computational resources. The payoff for a periodic activation of SRM is a window of opportunity during which spammers who are not yet blacklisted can send large amounts of Spam without being blocked.

In order to study the impact of black and white lists' update frequency on the accuracy of Spam filtering, we executed the reputation mechanisms in a batch mode with various update frequencies. The SRMs were executed each k minutes where k was set to 0.5, 1, 2, 5, 15, or 60 minutes. HDS parameters were: $w_0 = 60$ minutes, $n = 5$, and $T_{Pred} = 60$ minutes. In this experiment we used the BayesNet algorithm to train the classifiers for both the HDS-based SRMs and the EL-Based SRM.

Figures 3, and 2 depict the superiority of HDS-based SRMs also in batch mode. Looking at the predictability results (TPR), we see that the Heuristic SRM had the worst results, whereas HDS-based SRM (Spammingness) had the highest results among the SRMs. Figure 2 shows that for all update frequencies, the HDS-SRM (Spammingness) has a considerably higher AUC when compared to the Heuristic-based and EL-based SRMs. Since the AUC metric is an objective

performance metric that does not depend on a configurable threshold, it better reflects the superiority of the HDS over the other tested SRMs.

In general, all four SRMs' performance deteriorated, more or less at the same rate, as the black and white lists update frequency was reduced. This was well demonstrated by the drop in predictability, and AUC scores.

7.3 Exploiting the Historical Data

In this experiment we study two additional properties of the HDS-based SRM. First, we learn the effect of history length on the SRM classification performance. For this, we trained the SRM classifier using the BayesNet algorithm on HDS instances, where $w_0 = 15$ seconds, the number of historic windows, n, was varied between 1 to 14, and the prediction time T_{Pred} was fixed at 60 minutes. The black and white list were cleared once per a day. In the second experiment we study how the break of history data into exponential growing windows contribute to the overall classification performance. We used a fixed history size of 1,920 minutes and tested several (i.e., $n = 1 \ldots 6$) exponentially growing windows. In order to keep the same history length, the first window length was set to satisfy $w_0 * 2^{n-1} = 1,920$. The corresponding results are presented in Table 7.

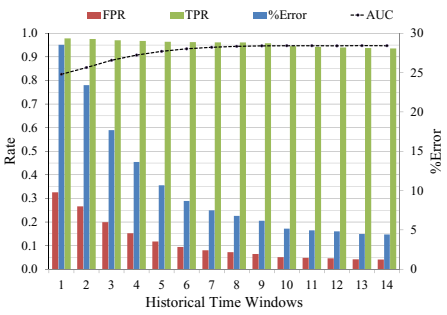


Fig. 4. The effect of history length on the SRMs' clssification performance

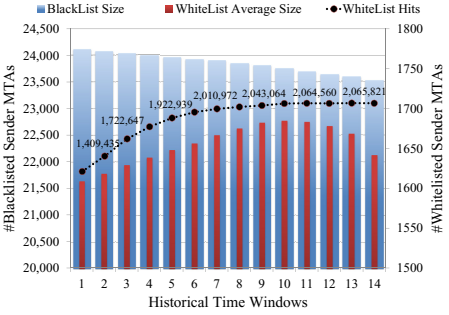


Fig. 5. The effect of history length on the black and white lists

The first experiment shows a mixed trend in the classification performance as a function of the number of historical time windows. The results presented in Figures 4 and 5 show that up to the ninth historical time-window the performance of the HDS-based SRMs improves, whereas, when using more historical time-windows, the AUC score ceased to improve, indicating a change of trend.

Due to the very small initial historical time window (i.e., only 15 seconds) we are able to notice the decrease in false positive and error rates as more time windows were used. Surprisingly, the true positive rate was also gradually decreased as the number of historical time windows grew. This can be explained by both the decrease in the blacklist size and the growing number of features that added

additional dimensions to the machine-learning problems, and therefore, made it more and more complex to learn from (a.k.a. "the course of dimensionality"). Notice that the FPR decreased faster then the TRP as the number of historical time windows increased.

Table 7. The effect of number of historical time windows (history = 960 min.)

	Number of windows					
Metric	1	2	3	4	5	6
%Error	8.20	6.79	6.50	6.41	6.39	6.44
TPR	0.334	0.444	0.473	0.480	0.482	0.495
FPR	0.003	0.002	0.002	0.002	0.002	0.005
AUC	0.666	0.721	0.735	0.739	0.740	0.745
BL Hits	17.2k	20.6k	22.3k	22.8k	22.9k	25.4k

In contrast to the black list, the white list average size increased until the tenth time window, and then the trend was reversed, were it begun an accelerated decline in size. Interestingly, the number of white-list hits remained more or less stable from the 9th to the 14th time windows, even when the white list average size dropped. The results in Table 7 show that under the same history length, the HDS-Based SRM perform better when breaking the history into exponentially growing time windows. With the second window (first history break) the classification AUC improved by 7.5%, in comparison to the AUC achieved when learning from the entire history at once. The AUC continued improving as the number of time windows grew, indicating that breaking the history into exponential growing time windows is a good idea, as the first time windows captures the fast changing behavior, while the larger windows captures longer trends, which are also important.

8 Discussion and Future Work

Experiment results presented in this paper show that aggregating behavior of MTAs over time is an effective way to elicit valuable information from email logs. The proposed method was found to be more effective than email-log-based and heuristic-based SRMs, tested under the *exact same conditions*. In fact, the ISP whose dataset we evaluated uses a blacklisting method similar to the Heuristic SRM, while the Email-log-based SRM is similar to state-of-the-art methods.

The same machine learning algorithms applied on HDS produce much more effective models than if applied on the non-aggregated data extracted from the raw email logs. The best results were obtained using HDS-based (Erraticness) SRM with nine time windows: AUC 0.907, TPR 90.7%, and FPR 0.4%. To the best of our knowledge these results are better than previously reported SRMs evaluated on data sets of similar scale. Another interesting fact is that HDS-based SRM blacklisted roughly the same number of IPs as the EL-based SRM, while incurring, by far, fewer classifications errors.

On our data set the upgrading of EL-based SRM to HDS-based SRM resulted in an elevated performance. Despite the differences in the particular features, we believe that other sender reputation mechanisms can also benefit from applying our method, i.e., aggregating historical email sending data over multiple exponentially growing time windows. We suggest that our results are general enough to motivate "upgrading" any EL to HDS.

Our experiments revealed a clear trade-off between batch size and the effectiveness of Spam filtering, when practicing a periodical execution of SRM. A less frequent execution of SRM resulted in less predictability power, as expected. The main reason for inefficient blacklisting when sender reputation is computed once in a long time period is the tendency of Spamming bots to send a number of Spam emails during a very short time period and go silent afterward [14].

Lastly, in our experiments the black and the white lists had on average 457,120 and 1,916,636 hits respectively. Each hit corresponded to a Spam or benign MTA's email that was not put through a content-based filter. Therefore, the ISP's filtering workload was decreased by: $\frac{black\ and\ white\ lists\ hits}{emails\ in\ TestSet} = \frac{2{,}373{,}756}{2{,}864{,}208} =$ 82.7%. This means that more than 4 out of 5 emails had hit one of the lists, and therefore, skipped the content-based filter, thanks to the HDS-RM black and white listing. This is a very significant contribution for ISPs, as their entire filtering workload, and consequently the energy consumed during the email filtering process can significantly be reduced.

As a future work we should further compare the HDS with some related works on data sets from various domains.

References

1. Alperovitch, D., Judge, P., Krasser, S.: Taxonomy of email reputation systems. In: ICDCS Workshops 2007 (2007)
2. Balthrop, J., Forrest, S., Newman, M.E.J., Williamson, M.M.: Technological networks and the spread of computer viruses. Science 304(5670), 527–529 (2004)
3. Beverly, R., Sollins, K.: Exploiting the transport-level characteristics of am. In: 5th Conference on Email and Anti-Spam, CEAS (2008)
4. Boykin, P., Roychowdhury, V.: Leveraging social networks to fight spam. IEEE Computer 38(4), 61–68 (2005)
5. Eleven. expurgate (June 2010), `http://www.eleven.de/overview-antispam.html`
6. Golbeck, J., Hendler, J.: Reputation network analysis for email filtering. In: First Conference on Email and Anti-Spam, Mountain View, California, USA (2004)
7. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explorations 11(1), 10–18 (2009)
8. Hao, S., Syed, N.A., Feamster, N., Gray, A.G., Krasser, S.: Detecting spammers with snare: Spatiotemporal network-level automated reputation engine. In: 18th USENIX Security Symposium (2009)
9. Koprinska, I., Poon, J., Clark, J., Chan, J.: Learning to classify email. Inf. Sci. 177(10), 2167–2187 (2007)
10. Liu, W.: Identifying and addressing rogue servers in countering internet email misuse. In: IEEE SADFE, pp. 13–24 (2010)
11. McAfee. Trustedsource, `http://www.trustedsource.org/`
12. Namestnikova, M.: Securelist spam report (December 2011), `http://www.securelist.com/en/analysis/204792212/Spam_report_December_2011`
13. Qian, Z., Mao, Z.M., Xie, Y., Yu, F.: On network-level clusters for spam detection. In: NDSS (2010)
14. Ramachandran, A., Feamster, N.: Understanding the network-level behavior of spammers. In: ACM SIGCOMM, Pisa, Italy (2006)

15. Ramachandran, A., Feamster, N., Vempala, S.: Filtering spam with behavioral blacklisting. In: ACM CCS, pp. 342–351 (2007)
16. Ruiz-Sanchez, M.A., Biersack, E.W.: Survey and taxonomy of ip address lookup algorithms. IEEE Network 15(2), 8–23 (2001)
17. Soldo, F., Le, A., Markopoulou, A.: Predictive blacklisting as an implicit recommendation system. In: IEEE INFOCOM, pp. 1–9 (2010)
18. SORBS, `http://www.au.sorbs.net/`
19. SpamCop, `http://www.spamcop.net/bl.shtml`
20. Spamhaus, `http://www.spamhaus.org`
21. Tang, Y., Krasser, S., Judge, P., Zhang, Y.-Q.: Fast and effective spam sender detection with granular svm on highly imbalanced mail server havior data. In: CollaborateCom, Atlanta, Georgia, USA (2006)
22. West, A.G., Aviv, A.J., Chang, J., Lee, I.: Preventing malicious behavior using spatio-temporal reputation. In: ACM EUROSYS 2010 (2010)
23. Youn, S., McLeod, D.: Improved spam filtering by extraction of information from text embedded image email. In: ACM SAC, New York, USA, pp. 1754–1755 (2009)

Operating System Kernel Data Disambiguation to Support Security Analysis

Amani S. Ibrahim, John Grundy, James Hamlyn-Harris, and Mohamed Almorsy

Centre for Computing and Engineering Software Systems
Swinburne University of Technology
Melbourne, Australia
{aibrahim,jgrundy,jhamlynharris,malmorsy}@swin.edu.au

Abstract. It is very challenging to verify the integrity of Operating System (OS) kernel data because of its complex layout. In this paper, we address the problem of systematically generating an accurate kernel data definition for OSes without any prior knowledge of the OS kernel data. This definition accurately reflects the kernel data layout by resolving the pointer-based relations ambiguities between kernel data, in order to support systemic kernel data integrity checking. We generate this definition by performing static points-to analysis on the kernel's source code. We have designed a new points-to analysis algorithm and have implemented a prototype of our system. We have performed several experiments with real-world applications and OSes to prove the scalability and effectiveness of our approach for OS security applications.

Keywords: Points-to Analysis, Operating System, Kernel Data Structures.

1 Introduction

Kernel data rootkits have the ability to alter the overall behavior of Operating Systems (OSes) – without injecting any malicious code into the kernel address space – by manipulating the pointer-based relations between kernel data structures. It is challenging to verify the integrity of kernel data, where an OS kernel contains thousands of data structures that have direct and indirect relations between each other with no explicit integrity constraints. In Windows and Linux OSes, from our analysis, nearly 40% of the inter-data structure relations are pointer-based relations (indirect relations), and 35% of these pointer-based relations are generic pointers (*e.g.* null pointers that do not have values, and void pointers that do not have associated type declarations in the source code). Such generic pointers get their values or type definitions only at runtime according to the different calling contexts. This makes kernel data a rich target for potent rootkits that exploit the points-to relations between data structures instances to hide/modify system runtime objects *e.g.* processes and threads.

Checking the integrity of OS's kernel data has been a major concern in many OS kernel security researches. However, current research efforts and practices [1-3] are severely limited as they depend on their prior knowledge of kernel data layout to manually resolve the ambiguous pointer-based relations, and thus they only cover 28%

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 263–276, 2012.

of kernel data structures (as discussed by Carbone *et al.* [4]) that relate to well-known objects. Current approaches also do not consider the generic pointer relations between structures, making their approach imprecise and vulnerable to wide range of attacks that can exploit these pointers. This results in security holes, limited protection and an inability to detect zero-day threats, and raises the need to obtain an accurate kernel data definition that resolves pointer-based relation ambiguities. Such a definition is an important step in implementing different systematic OS security applications *e.g.* memory forensics tools, virtual machine introspection (VMI), dynamic object uncovering and kernel integrity checking tools.

In this paper, we address the problem of systematically building an accurate kernel data definition that precisely models data structures, reflects both direct and indirect relations, and generates constraint sets between structures. We extend KDD (Kernel Data Disambiguator), a tool that can generate a sound kernel data definition for any C-based OS (*e.g.* Windows, Linux, UNIX) without any prior knowledge of the OS kernel data layout [19]. KDD disambiguates pointer-based relations including generic pointers – to infer their candidate types/values – by performing static *points-to* analysis on the kernel's source code. *Points-to* analysis is the problem of determining statically a set of locations to which a given variable may point to at runtime. *Points-to* analysis for C programs has been widely used in compiler optimization, memory error detection and program understanding [5,6]. However, none of these approaches meet our requirements in analyzing the kernel as they do not scale to the enormous size and complexity typical of an OS kernel. They also typically sacrifice precision for performance. In KDD, precision is an important factor. We want the most precise points-to sets to be computed. To meet our requirements, we implemented a new *points-to* analysis algorithm that has the ability to provide interprocedural, context-sensitive, field-sensitive and inclusion-based *points-to* analysis for large programs that contain millions lines of code *e.g.* OS kernel. We have implemented a prototype system of KDD, and performed several experiments on KDD using large-scale real-world applications including OSes to prove its effectiveness and scalability.

Section 2 presents the motivation for our work and key related work. Section 3 presents KDD's approach. We discuss implementation and evaluation details in Section 4. Finally we discuss results and draw key conclusions.

2 Background

Ensuring reliability of large systems *e.g.* OSes is a difficult problem, especially C-based ones. C-based OSes use C structures heavily to model objects. They also use pointers extensively to simulate call-by-reference semantics, emulate object-oriented dispatch *via* function pointers, avoid expensive copying of large objects, implement lists, trees and other complex data structures, and also as references to objects allocated dynamically on the heap [7]. Moreover, objects can be cast to multiple types during their lifetime, and a pointer deposited in a field under one object may be read from a field under another object. This makes the analysis of kernel's data structures a non-trivial task, further complicated by the fact that data structures are implementation-dependent. Hence, imprecise points-to analysis will therefore result in improper

assumptions about the indirect relations between structures. As C allows casting, values can be copied from a pointer to a non-pointer and vice versa, points-to sets should be computed to all program variables, not just declared pointers.

To get a concrete idea of the pointers problem in OSes, we discuss three problems of generic pointers we need to address. First, use of void pointers; void pointers support a form of polymorphism. At runtime, if a pointer is not void, its target object should have the type of a pointer. However, the problem with use of 'void' type is that the target object type(s) can only be identified at runtime. The wide use of such void pointers hinders performing systematic integrity checks on kernel data, where there are no type constraints for void *. This assists hackers in exploiting these pointers to point to somewhere else in memory. Second, use of null pointers; these are used for example to implement linked lists (e.g. single, doubly or triply) which are heavily used in OSes to maintain running objects. The C definition makes a linked list points to a linked list, but actually during runtime it points to a specific object type according to the calling contexts. The problem is that the objects structured in a list can be recognized only during runtime (*e.g.* object type, number of running instances and locations). Thus, null pointers manipulation helps hackers to hide or change runtime objects. Identifying the object type that a list may hold at the offline analysis phase helps significantly in identifying a set of constraints on the runtime objects to detect invalid pointer manipulations. Third, use of casting; C data types can be subverted by casting. A pointer of a given type can be cast to point to any other C type. A major problem with casts is that they induce relationships between objects that appear to be unrelated, enabling hackers to exploit data structure layout in physical memory.

Kernel data integrity checking has been studied intensively [8,9,2,1]. However, these research efforts are limited to the OS expert knowledge to resolve the ambiguous pointer relations. OSck [8] and SigGraph [10] follow a systematic approach to resolve the pointer-based relation, however they do not solve the generic pointers (indirect relations) problem. To the best of our knowledge, KOP [4] (a Microsoft internal tool) , is the first and only tool that employed points-to analysis in order to analyze an OS kernel to solve generic pointers ambiguities. However, KOP is limited. It uses a medium-level intermediate representation (MIR) that complicates the analysis and results in improper points-to sets. MIR is extremely big in size, omits very important information such as declarations, data types and type casting, and creates a lot of temporary variables that are allocated identically to source code variables and thus are not easily distinguishable from source code variables [11]. Also in KOP, the points-to sets of the void * objects are not precise and thus they use a set of constraint criteria (OS-specific) at runtime to find out the appropriate candidate for the object.

Many state-of-the-art tools have been developed for points-to analysis of C programs [5,6,12]. Their use has predominantly been for compiler optimizations and program understanding, and their main goal has thus been performance. They differ mainly in how they group alias information. There are two main algorithms used to group alias information: *Andersen's* [13] and *Steensgaard's* [14]. Fig. 1 shows a C code fragment and the points-to sets computed by those algorithms. Andersen's is the slowest but the most precise while Steensgaard's is the fastest but is imprecise. Anderson's approach creates a node for each variable and the node may have different

edges. Steensgaard's algorithm groups alias sets in one node and each node has one edge. Based on these approaches there are different types of analysis that trade-off performance and precision: *(i) Field-Sensitivity*; distinguishing the different fields inside structures and unions *i.e.* each field has a distinct points-to set. *(ii) Context-Sensitivity*; distinguishing objects created through different call sites. Context-sensitive algorithms are more precise, but are much slower in performance and complicated to implement. *(iii) Flow-Sensitivity;* considers the effects of pointer assignments with respect to the call-graph. *(iv) Inclusion-Based;* considers dependency relations between structures to represent the inclusion constraints.

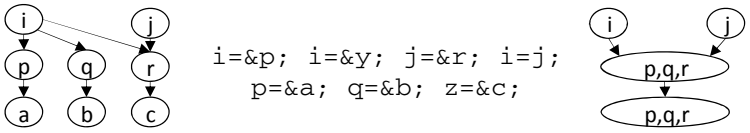


Fig. 1. Alias information grouping by Steensgaard's and Andersen's approaches

A number of research efforts have attempted performing field and context sensitivity analysis on large programs [15,6,16,17]. However none have been shown to scale to large programs *e.g.* OS's kernel code with a high rate of precision. Also, these algorithms are used during program compilation time to name objects by allocation site, not by access path. Thus, they do not enable solving the ambiguity of null pointers.

3 Our Approach

KDD takes the source code of an OS kernel as input and outputs an accurate directed type-graph that represents the kernel data definition. This type-graph summarizes the different data types located in the kernel along with their connectivity patterns and reflects the inclusion-based relations between kernel data structures for both direct and indirect relations. A high-level representation of this analysis process is shown in Fig. 2. To facilitate the analysis, we use Abstract Syntax Tree (AST) as a high-level intermediate representation for the source code. The AST captures the essential structure of the code that reflects its semantic structure while omitting unnecessary syntactic details. Since it has been established that flow-sensitivity does not add significant precision over a flow-insensitive when we ignore the control-flow of programs [18], we consider flow-insensitive points-to analysis in KDD.

Two main phases of the analysis are used to build the type-graph. The first analysis step is straightforward, and its goal is computing the direct relations between kernel data structures that have clear type definitions. This is done by performing a compiler-pass approach on the AST files to extract the data structure type definitions by looking for *typedef* aliases, and extract their fields with the corresponding type definition. Nodes are data structures and edges are data members of the structures. The second step is the most important step and its goal is computing the indirect relations between structures. Indirect relations (generic pointer dereferencing) cannot be computed from the AST

directly. To solve this problem, we have developed a new points-to analysis algorithm to statically analyze the kernel's source code to get an approximation for every generic pointer dereferencing based on Anderson's approach. We consider all forms of assignments and function calls. Data structures are flattened to a scalar field. Type casting is handled by inferring locations accessed by the pointer being cast. Kernel objects are represented by their allocation site according to the calling contexts. The target of this step is a graph *G (N, E)*, where *N* is the set of nodes representing global and local variables, fields, array elements, procedure arguments\parameters and function returns. *E* is a set of directed edges across nodes representing, assignments and function calls. The graph has different types of nodes and edges (details omitted for brevity - for more details please see [19]). The type-graph of this step is created and refined by our points-to analysis algorithm in three steps, discussed below.

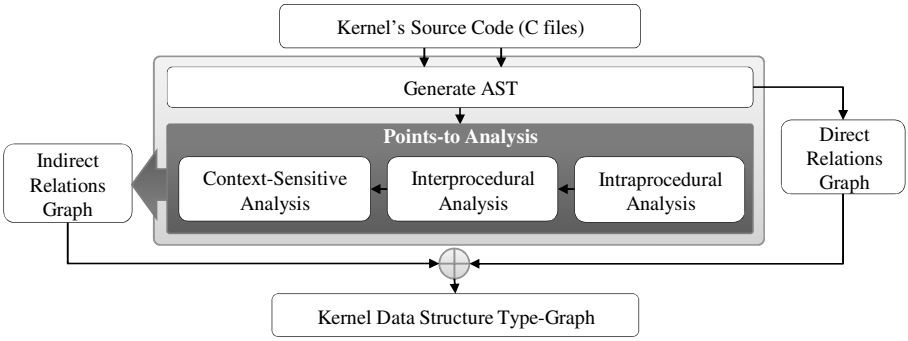


Fig. 2. High-level view of KDD operation

3.1 Intraprocedural Analysis

The goal of this analysis step is to compute a local type-graph but without information about caller or callees. Algorithm 1 summarizes our intraprocedural analysis algorithm. KDD takes the AST file as input and outputs an initial graph that contains nodes, as follows: *(i) Variables*; create node for each variable declaration and check the function scope to find out if it is a local or global variable. *(ii) Procedure definition;* create node for each formal-in parameter. *(iii) Procedure call*; create nodes for each formal-in argument, in addition to a dummy node for each formal-in argument represented by its relative position (index) in the procedure. These dummy nodes will be used later to create an implicit assignment relation between the formal-in arguments and formal-in parameters. For example, given *G(x, y)*, we create two nodes for *x* and *y* and other two dummy nodes *G:1* and *G:2*. *(iv) Assignments;* create nodes for the left and right hand sides. *(v) Function return;* create one node for the return statement itself and one for the returned value. Meanwhile, KDD builds the initial edges by computing the *transfer function* (TF) as described in table 1. TF is a formal description for the relation between the nodes created for each of the previous entities.

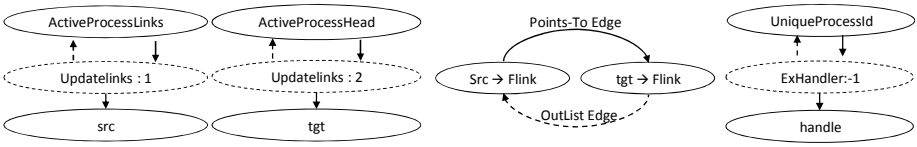
Algorithm 1. Intraprocedural Analysis Algorithm

```
1:  Procedure IntraproceduralAnalysis (ASTFile F)
2:      ∀ ASTLine L ∈ F
3:      if L ∋ variable V declaration statement then check function scope;
4:          if (scope == null) then V ⊆ global variable
5:          elseif L ∋ function parameters then V ⊆ Local function parameter
6:          else V ⊆ Local variable
7:          Create node();
8:      endif
9:      if L ∋ assignment | function call | return statement then Compute transfer function ();
10: end
```

Table 1. Transfer function description; local points-to sets *pts()*, constraints between nodes, and edges (→ a directed *inlist* edge between two nodes, ← a directed *outlist* edge).

	Code	Local pts()	Constraints	Edges
Proc	***Description**;* relation between formal-in parameters and the dummy nodes that hold the indexes of the parameters. ***Edges**; inlist* edge between each formal-in parameter node and its relevant dummy node, and *outlist* edge from the dummy node to its relevant formal-in parameter node.			
	proc(p)	*pts (proc:1) ⊇ pts(p)*	*proc:1 ⊇ p*	*proc:1 → p, proc:1 ← p*
Assignment	***Description**;* relation between left and right hand sides (HSs) of the assignment statement. ***Edges***; *inlist* edge from left HS to right HS, and *outlist* edge from the right HS to left HS.			
	p=&q	*loc (q) ∈pts(p)*	*p ⊇[q]*	*p → q, p ← q*
	p=q	*pts (p) ⊇pts(q)*	*p ⊇q*	*p → q, p ← q*
	*p=*q*	*∀v ∈ pts(q) : pts (p) ⊇ pts(v)*	*p ⊇ *q*	*p → *q → v, p ← *q ← v*
	**p=q*	*∀v ∈ pts(p) : pts (v) ⊇ pts(q)*	**p ⊇q*	*v → *p → q, v ← *p ← q*
Call	***Description**;* relation between the formal-in arguments nodes and dummy nodes. ***Edges***; *inlist* edge between each argument node and its relevant dummy node.			
	proc(q);	*pts(q) ⊇pts (proc:1)*	*q ⊇proc:1*	*q → proc:1*
Return	***Description***; relation among left hand side, the procedure return node and the returned value node. ***Edges***; *inlist* edge between the left hand side and the return node, *inlist* edge between the return node and retuned value node and *outlist* edge between the return node and the left hand side.			
	p = fn() return q;	*pts (p) ⊇pts(q)*	*p ⊇q*	*p → q*

Consider a call to a procedure called "*Updatelinks"*, where the formal-in parameters are *(src, tgt)*, and the actual passed arguments are (&ActiveProcessLinks, &ActiveProcessHead), and consider these explicit assignment statements *(src→Flink = tgt→Flink; tgt→Blink = src→Blink).* KDD computes the TF for those statements as shown in Fig. 3(a) and Fig. 3(b), respectively. For the *return*, given this fragment of code UniqueThreadId = ExHandler(), the computed TF is shown in Fig. 3(c).



(a) (b) (c)

Fig. 3. Intraprocedural analysis graph

3.2 Interprocedural Analysis

In this phase we perform an interprocedural analysis that enables performing points-to analysis across different files to perform whole-program analysis. The result of this phase of analysis is a graph that computes the calling effects (returns, arguments and parameters), but without any calling context information yet. This is done by propagating the local points-to sets computed at the intraprocedural step to their use sites consistently with argument index in the call site. Thus we can map between the procedure arguments and parameters. Fig. 4 shows the analysis results of this step for the examples discussed in the intraprocedural analysis step. Algorithm 2 summarizes this interprocedural analysis step.

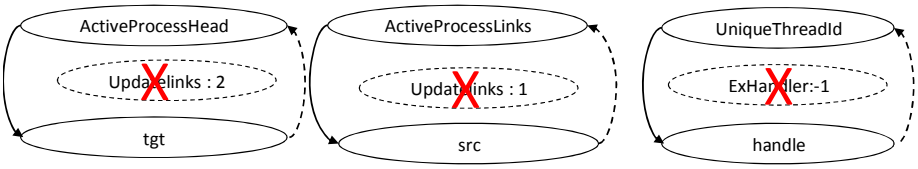


Fig. 4. Interprocedural analysis result

Algorithm 2. Intraprocedural Analysis Algorithm

```
1: Procedure Interprocedural Analysis (Graph G)
2:     ∀ Node N ∈ G
3:         if ∃ N has the form N(Procedure Name : index) then
4:             Create inlist edge (N.outlist, N.inlist); Create outlist edge (N.intlist, N.outlist);
6:             Delete dummy nodes ();
7:         end if
8: end
```

3.3 Context-Sensitive Points-to Analysis

The key in achieving context-sensitivity is to obtain the return of procedures according to the given arguments combined with the call site. Algorithm 3 summarizes our context-sensitive analysis of this step, performed in three sub-steps, as follows:

Points-to Analysis. A well-known complication in this analysis is the order of which nodes will be analyzed first, where this can greatly affect performance. A good choice is to analyze nodes in a topological order [12], by building a Procedure Dependency Graph (PDG). Our PDG consists of nodes representing the statements of the data dependency in the program. Data dependency between two statement nodes exists if a variable at one statement might reach the usage of the other variable at another statement. We start with the top node (according to the computed PDG) that does not have any dependencies, and thus we guarantee that each node has its *inlist* nodes already analyzed before proceeding with the node itself. We expand the local dereferencing of the pointers to get the points-to relations between the caller and callee. Then we propagate the points-to set of each node into its successors accumulating to the bottom node.

For the acyclic points-to relations, pointers are analyzed iteratively until their points-to sets are fully traversed. For recursions, we analyze pointers in each recursion cycle individually to make the analysis algorithm accommodates to modification and read effects introduced by the calls.

Algorithm 3. Points-to Analysis

```
1:  Procedure PointsToAnalysis (PDG PDG, Graph G, TransferFunction TF)
2:      ∀ Node N ∈ G
3:          ∀ InListNode in ∈ N.InList
4:              Compute points-to set (in);   N. PointstoSet. Add (in. PointstoSet);
5:          N. PointstoSet. Add (in);
6:          ∀ PointedToNode toN ∈ N. PointstoSet
7:              ∀ Child ch ∈ N. Children
8:                  CopyNode (ch);   Connect edges ();
9:          UpdateNodePointsTo (N, toN); Write the Graph();
10: end procedure
11: Procedure UpdateNodePointsTo (Node N, PointedToNode toN)
12:     if N.fnScope != toN.fnscope) then ∀ SubPointedToNode StoN ∈ toN. PointstoSet
13:         if StoN.fnScope == N.fnScope then N. PointstoSet. Add(StoN);
14:     else UpdateNodePointsTo (N, toN);
15: end procedure
```

Graph Unification. This step targets to compute a consistent graph. Consider the following procedure and procedure call: `void updatelinks(PList_Entry src, PList_Entry tgt)` and `Updatelinks(&ptr->ActiveProcessLinks, &ActiveProcessHead)`. We pass an object type to the procedure; however *Update-links* manipulates the fields of passed object *Flink* and *Blink*. As the definition of the _PList_Entry data type is: typedef struct _LIST_ENTRY { struct _List_Entry *Flink; struct _List_Entry *Blink;} List_Entry, *PList_Entry;

To solve this problem, we apply a unification algorithm to the type-graph, as follows: given node A with points-to set S and $T \in S$, if T has *child-relation* edge with f; we create a *points-to* edge between f and A. Fig. 5(a) shows the analysis result of this step of this example piece of code.

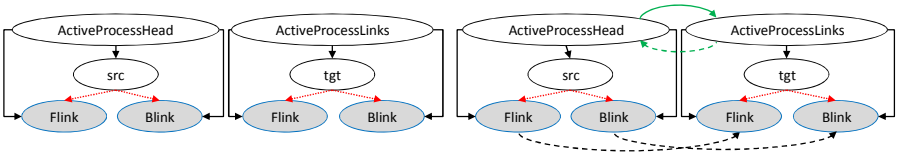


Fig. 5. Context-Sensitive Analysis

Context-Sensitivity. Without context-sensitivity, the analysis of functions that have different calling context would result in very general points-to sets for their arguments. To achieve context-sensitivity, we used the transfer function for each procedure call and apply its calling contexts, to bind the output of the procedure call according to the calling site. The points-to edge here is a tuple $\langle n, v, c \rangle$ represents a pointer n points to variable v at context c, where the context is defined by a sequence

of functions and their call-sites to find out valid call paths between nodes. Performing context-sensitive analysis solves two problems: the calling context and the indirect (implicit) relations between nodes. These indirect relations are calculated for each two nodes that are in the same function scope but not included in one points-to set. Such that, ∀ two nodes v and n where $v \in pts(n)$ and v and n has different function scope, check the function scope of n and x where $x \in pts(v)$, if the function scope is the same then create a *points-to* edge between n and x. Fig. 5(b) shows the final context-sensitive analysis for the *Updatelinks* example. We discovered that there is an indirect *points-to* relation from *ActiveProcessHead* to *ActiveProcessLinks*.

Finally, we write the type-graph. We replace each variable node with its data type and for fields and array elements we add the declared parent type. Finally, we format the results of our analysis to the DOT language, as a simple visualization for the kernel data layout to be used by the other OS security solutions that make use of KDD.

4 Implementation and Evaluation

We have implemented a prototype of KDD using C#. KDD uses *pycparser* [20] to generate AST files of the kernel's source code. KDD then uses the AST files to applying our points-to analysis algorithm to generate the type-graph. We have used Microsoft's Parallel Extensions to leverage multicore processors in an efficient and scalable manner to implement KDD. Threading has also been used to improve parallelization of computations (.Net supports up to 32768 threads on a 64bit platform). In the intraprocedural analysis, KDD analyzes each AST file using a separate thread. For interprocedural analysis, KDD allocates a thread for each procedure to parses the AST files to map between the procedure parameters and arguments. However, for the context-sensitive, the analysis is done on sequential-basis as each node depends on its predecessors.

We performed three types of experiments with KDD to demonstrate its scalability and effectiveness. We measured the soundness and precision of KDD using different sets of benchmarks. We analyzed the Linux kernel v3.0.22 and WRK ((Windows Research Kernel) using KDD, and performed a comparison between the computed pointer relations using KDD, and the manual efforts to solve these relations in both kernels. We implemented a memory mapping tool that uses our type-graph to correctly map the physical memory bytes from an introspected virtual machine to actual runtime objects, in order to prove KDD efficiency in defining the kernel data. Our implementation and evaluation platform is 2.5GHz core i5 processor with 12 GB RAM.

4.1 Soundness and Precision

The points-to analysis algorithm is *sound* if the points-to set for each variable contains all its actual runtime targets, and is *imprecise* if the inferred set is larger than necessary. Imprecise results could be sound *e.g.* if $pts(p) = \{a,c,b\}$ while the actual runtime targets are a and b, then the algorithm is sound but not precise and thus there exist false positives. If $pts(p) = \{a,c\}$ and the actual runtime targets are a and b then the algorithm is not sound nor precise, and thus there exist false positives and negatives. KDD is sound as it performs the points-to analysis on all program variables not just declared pointers, in order to cover all runtime targets whilst omitting unnecessary local variables.

We used a selection of C programs from the SPEC2000 and SPEC2006 benchmark suites, and other open source C programs, to measure the soundness and precision of KDD. Table 2 shows the characteristics of these benchmark C programs, in addition to the precision of the KDD analysis of each. We also show indications of memory, time and processor usage of running KDD on these benchmark programs. We manually verified each program to get an accurate estimation of the points-to set. For programs that are less than 4 KLOC, we instrumented pointers manually. For larger programs we picked a random set of generic pointers based on our understanding of the program. However, we could not measure precision for some programs because of their size. We also ran each program and monitored allocations in physical memory to get the actual runtime targets (i.e. relevant points-to set). Then, we used the equation below to calculate precision. Our results show that for the benchmark C programs analyzed by KDD, we achieved a high level of precision and 100% of soundness. The results also show that for significantly sized C programs KDD is able to process the application code with very acceptable CPU time and memory usage.

$$Precision = \frac{Relevant\ Points\ to\ Set\ \cap\ Retrieved\ Points\ to\ Set}{Retrieved\ Points\ to\ Set}$$

Table 2. Soundness and Precision Results running KDD on a suite of benchmark C programs. LOC is lines of code. Pointer Inst is number of pointer instructions. Proc is number of Procedure definitions. Struct is number of C struct type definitions. AST T is time consumed to generate the AST files, AST M is memory usage, and AST C is CPU usage. TG T is time consumed to build the type-graph, TG M is memory usage, TG C is CPU usage.

Benchmark	LOC (K)	Pointer Inst	Proc	Struct	ASTt (sec)	ASTm (MB)	ASTc (%)	TGt (sec)	TGm (MB)	TGc (%)	P (%)
art	1.2	286	43	19	22.7	21.5	19.9	73.3	12.3	17.6	100
equake	1.5	485	40	15	27.5	25.4	20.4	87.5	14.1	21.1	98.6
mcf	2.4	453	42	22	43.2	41	28.5	14	23	27	97.2
gzip	8.6	991	90	340	154.2	144.6	70.5	503.3	81.4	68.3	95.1
parser	11.3	3872	356	145	305.2	191.2	76.7	661.4	107.8	74.3	94.5
vpr	17.7	4592	228	398	316.1	298.7	80.2	1031.5	163.2	79	NA
gcc	222.1	98384	1829	2806	3960.5	3756.5	93.5	12962	2200	94	NA
sendmail	113.2	9424	1005	901	2017.2	1915.1	91.6	6609	1075.0	91.5	NA
bzip2	4.6	759	90	14	82.3	78.1	45.5	271.6	44.2	42.9	95.9

4.2 Kernel Analysis

To illustrate the scale of the problem presented by C-based OSes, we performed a simple statistical analysis on the WRK (~ 3.5 million LOC) and Linux kernel v3.0.22 (~ 6 million LOC) to compute the amount of type definitions (data structures), global variables and generic * used in their source code. Table 3 summarizes this analysis.

KDD scales to the very large size of such OSes. KDD needed around 28 hours to analyze the WRK and around 46 hours to analysis the Linux kernel. Comparing KDD to KOP, KOP has to be run on a machine with 32 GB RAM and needed around 48 hours to analyze the Windows Vista kernel. The performance of KDD could be improved by increasing RAM and processing capabilities. As our points-to analysis is performed offline and just once for each kernel version, the performance overhead of analyzing kernels is acceptable, and does not present a problem for any security

application that wants to make use of KDD's generated type graph. Re-generation of the graph is only necessary for different versions of a kernel where data structure layout changes may have occurred.

Table 3. Kernel source code analysis. 1st column shows the number of type definitions, 2nd column is the number of global variables, DL column shows the number of doubly linked lists and last column reflects the number of unsigned integers that represent casting problem. AST column shows AST files size in gigabyte.

	TD	GV	Void*	Null*	DL	Uint	AST
Linux	11249	24857	5424	6157	8327	4571	1.6
WRK	4747	1858	1691	2345	1316	2587	0.9

To evaluate the accuracy of KDD's generated OS type-graphs, we performed a comparison between the pointer relations inferred by KDD and the manual efforts of OS experts to solve these indirect relations in both kernels. We manually compared around 74 generic pointers from WRK and 65 from the Linux kernel. These comparisons show that KDD successfully deduced the candidate target type/value of these members with 100% soundness. Because of the huge size of the kernel, we could not measure its precision for nearly 60% of the members we used in our experiment, as there is no clear description for these members from any existing manual analysis. We were thus only able to measure precision for well-known objects that have been analyzed manually by security experts and whose purpose and function is well-known and documented. The resulting precision was around 96% in both kernel versions.

4.3 Object-Graph for Security Monitoring

We modified our earlier-developed kernel security monitoring tool, *CloudSec* [21], to use our KDD-generated type-graph to traverse the physical memory of a running OS from a hypervisor level, in order to construct a correct object-graph that identifies all the running instances of the data structures for a running Virtual Machine (VM). The objective of this experiment was to demonstrate the effectiveness of KDD in computing a precise kernel data definition, not to detect threats where we utilize a traditional memory traversal technique that is vulnerable to object hiding attacks. *CloudSec* is a security appliance that has the ability to monitor VMs' memory from outside the VM itself, without installing any security code inside the VM. *CloudSec* uses memory traversal techniques to map the running objects based on manual profiles that describe the direct and indirect relations between structures [21]. In this experiment, we used our KDD-generated type-graph to locate dynamic objects by traversing kernel memory starting from the OS global variables and then following pointer dereferencing until we covered all memory objects. We used *CloudSec* to map the physical memory of a VM running Windows XP 64bit. The performance overhead of *CloudSec* to construct the object-graph for the entire kernel running objects was around 6.3 minutes for a memory image of 4GB on a 2.8 GHz CPU with 6GB RAM. To evaluate the mapping results, we considered the global variable *PsActiveProcessHead* then followed pointer dereferencing until we covered 43 different data structures with their running instances. We compared the results with the internal OS view using Windows

Debugger. *CloudSec* successfully mapped and correctly identified the running kernel objects, with a low rate of false positives (~1.5% in traversing balanced trees). This demonstrates that, for these 43 data structures monitored, our generated type-graph is accurate enough for kernel data disambiguation to support security monitoring.

5 Discussion

KDD is a static analysis tool that operates offline on an OS kernel's source code to generate a robust type-graph for the kernel data that reflects both the direct and indirect relations between structures, models data structures and generates constraint sets on the relations between them. Our experiments with KDD have shown that the generated type-graph is accurate, and solves the null and void pointer problems with a high percentage of soundness and precision. KDD is able to scale to the enormous size of kernel code, unlike many other points-to analysis tools. This scalability and high performance was achieved by using AST as the basis for points-to analysis. The compact and syntax-free AST improves time and memory usage efficiency of the analysis. Instrumenting the AST is more efficient than instrumenting the machine code *e.g.* MIR because many intermediate computations are saved from hashing.

Performing static analysis on kernel source code to extract robust type definitions for the kernel data structures has several advantages: *(i) Systematic Security*; enables the implementation of systematic security solutions. By this we mean that we have the ability to systematically protect kernel data without the need to understand deep details about kernel data layout in memory, as is done to date. *(ii) Performance Overhead;* we minimize the performance overhead in security applications as a major part of the analysis process is done offline. If no static analysis were done, every pointer dereference would have to be instrumented, which increases performance overhead. *(iii) Detecting Zero-Day Threats;* we maximize the likelihood of detecting zero-day threats that target generic (*via* bad pointer dereferencing) or obscure kernel data structures. *(iv) Generating Robust Data Structures Signatures*; KDD generates robust data structure signatures that can be used by brute force scanning tools [10]. *(v) Type-Inference;* declared types of C variables are unreliable indications of how the variables are likely to be used. Type inference determines the actual type of objects by analyzing the usage of those objects in the code base. *(vi) Function Pointer Checking*; enable checking the integrity of kernel code function pointers that reside in dynamic kernel objects, by inferring the target candidate type for each function pointer. This decreases the need to instrument every function pointer during runtime, as the addresses of objects that hold these pointers change during runtime.

To the best of our knowledge, there is no similar research in the area of systematically defining the kernel data structure with the exception of KOP [4]. However in addition to the limitations discussed in the related work section, the points-to sets of KOP are not highly precise compared to KDD. This is because they depend on the Heintze points-to analysis algorithm [6], which is used in compilers for fast aliasing. In addition, KOP computes transitive closures in order to perform the points-to analysis - this increases the performance overhead of the analysis. To the best of our knowledge, our points-to analysis algorithm is the first points-to analysis technique that

depends on the AST to provide interprocedural, context and field sensitive analysis. Buss *et al.* [22] has an initiative in performing points-to analysis based on the AST of the source code. However, their algorithm is field and context insensitive.

6 Summary

The wide existence of generic pointers in OS kernels makes kernel data layout ambiguous and thus hinders current kernel data integrity research from providing the preemptive protection. In this paper, we described KDD, a new tool that generates a sound kernel data structure definition for any C-based OS, without prior knowledge of the OS kernel data layout. Our experiments with our prototype have shown that the generated type-graph is accurate and solves the generic pointer problem with a high rate of soundness and precision.

Acknowledgement. The authors are grateful to Swinburne University of Technology and FRST Software Process and Product Improvement project for support for this research.

References

1. Baliga, A., Ganapathy, V., Iftode, L.: Automatic Inference and Enforcement of Kernel Data Structure Invariants. In: Proc. of 2008 Annual Computer Security Applications Conference, pp. 77–86 (2008)
2. Dolan-Gavitt, B., Srivastava, A., Traynor, P., Giffin, J.: Robust signatures for kernel data structures. In: Proc. of 16th ACM Conference on Computer and Communications Security, Illinois, USA, pp. 566–577 (2009)
3. Ibrahim, A., Shouman, M., Faheem, H.: Surviving cyber warfare with a hybrid multiagent-base intrusion prevention system. IEEE Potentials 29(1), 32–40 (2010)
4. Carbone, M., Cui, W., Lu, L., Lee, W.: Mapping kernel objects to enable systematic integrity checking. In: Proc. of 16th ACM Conference on Computer and Communications Security, Chicago, USA, pp. 555–565 (2009)
5. Whaley, J., Lam, M.S.: Cloning-based context-sensitive pointer alias analysis using binary decision diagrams. In: Proc. of ACM SIGPLAN 2004 Conference on Programming Language Design and Implementation, Washington DC, USA, pp. 131–144 (2004)
6. Heintze, N., Tardieu, O.: Ultra-fast aliasing analysis using CLA: a million lines of C code in a second. In: Proc. of ACM SIGPLAN 2001 Conference on Programming Language Design and Implementation, Utah, USA, pp. 254–263 (2001)
7. Mock, M., Atkinson, D.C., Chambers, C., Eggers, S.J.: Program Slicing with Dynamic Points-To Sets. IEEE Trans. Softw. Eng. 31(8), 657–678 (2005)
8. Hofmann, O.S., Dunn, A.M., Kim, S.: Ensuring operating system kernel integrity with OSck. In: Proc. of 16th International Conference on Architectural Support for Programming Languages and Operating Systems, California, USA, pp. 279–290 (2011)
9. Petroni, N.L., Hicks, M.: Automated detection of persistent kernel control-flow attacks. In: Proc of 14th ACM Conference on Computer and Communications Security, Alexandria, Virginia, USA, pp. 103–115 (2007)

10. Lin, Z., Rhee, J., Zhang, X.: SigGraph: Brute Force Scanning of Kernel Data Structure Instances Using Graph-based Signatures. In: Proc. of 18th Network and Distributed System Security Symposium, San Diego, CA (2011)
11. Chen, Y., Venkatesan, R., Cary, M., Pang, R., Sinha, S., Jakubowski, M.H.: Oblivious Hashing: A Stealthy Software Integrity Verification Primitive. In: Petitcolas, F.A.P. (ed.) IH 2002. LNCS, vol. 2578, pp. 400–414. Springer, Heidelberg (2003)
12. Pearce, D.J., Kelly, P.H., Hankin, C.: Efficient field-sensitive pointer analysis for C. In: Proc. of 5th ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering, Washington DC, pp. 37–42. ACM (2004), 996835
13. Andersen, L.: Program Analysis and Specialization for the C Programming Language. University of Copenhagen (1994)
14. Steensgaard, B.: Points-to analysis in almost linear time. In: Proc. of 23rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, Florida, United States, pp. 32–41 (1996)
15. Hardekopf, B., Lin, C.: The ant and the grasshopper: fast and accurate pointer analysis for millions of lines of code. In: Proc. of 2007 ACM SIGPLAN Conference on Programming Language Design and Implementation, California, USA, pp. 290–299. ACM (2007), 1250767
16. Yu, H., Xue, J., Huo, W., Feng, X., Zhang, Z.: Level by level: making flow- and context-sensitive pointer analysis scalable for millions of lines of code. In: Proc. of 8th Annual IEEE/ACM International Symposium on Code Generation and Optimization, Ontario, Canada, pp. 218–229 (2010)
17. Lattner, C., Lenharth, A., Adve, V.: Making context-sensitive points-to analysis with heap cloning practical for the real world. In: Proc. of 2007 ACM SIGPLAN Conference on Programming Language Design and Implementation, California, USA, pp. 278–289 (2007)
18. Hind, M., Pioli, A.: Which pointer analysis should I use? In: Proc. of 2000 ACM SIGSOFT International Symposium on Software Testing and Analysis, Portland, Oregon, United States, pp. 113–123 (2000)
19. Ibrahim, A.S., Grundy, J.C., Hamlyn-Harris, J., Almorsy, M.: Supporting Operating System Kernel Data Disambiguation using Points-to Analysis. In: Proc. of 27th IEEE/ACM International Conference on Automated Software Engineering, Essen, Germany (2012)
20. Bendersky, E.: pycparser: C parser and AST generator written in Python (2011), http://code.google.com/p/pycparser/
21. Ibrahim, A.S., Hamlyn-Harris, J., Grundy, J., Almorsy, M.: CloudSec: A Security Monitoring Appliance for Virtual Machines in the IaaS Cloud Model. In: Proc. of 2011 International Conference on Network and System Security (NSS 2011), Milan, Italy (2011)
22. Buss, M., Edwards, S.A., Bin, Y., Waddington, D.: Pointer analysis for source-to-source transformations. In: Proc. of 5th IEEE International Workshop on Source Code Analysis and Manipulation, September 30-October 1, pp. 139–148 (2005)

FlexCOS: An Open Smartcard Platform for Research and Education

Kristian Beilke and Volker Roth

Freie Universität Berlin, Berlin, Germany
kristian.beilke@fu-berlin.de
http://www.inf.fu-berlin.de/groups/ag-si/

Abstract. The smartcard industry treats their know-how and products as confidential. Consequently it is difficult to do research on smartcards without NDAs. We present a platform meant to lower the barrier to entry for smartcard research. It is built with mostly free software on affordable hardware and is designed to provide access to every level of the system. This platform allows to do research without NDAs as well as a more practical approach to education, since it is reproducible with limited costs. Its main advantages are the ability to make changes to the operating system, debugging on the running platform, and the possibility to integrate new hardware components into the system. We achieve this by compromising on hardware security aspects and form factor, and focus on the software aspects instead.

Keywords: Smartcards, FPGA, Embedded Systems.

1 Introduction

Smartcards are one of the most widely spread embedded systems. Mobile phones, banking cards, access cards and more recently also electronic ID-documents use them as integral components. Smartcards are mass-produced and have a low purchase price, even in small quantities, and they represent a fascinating combination of hardware and software design optimized for security. Hardly any other system combines these properties, none of which is comparably ubiquitous. It is thus surprising that, compared to other embedded systems, smartcards are underrepresented in research and education. A primary reason for this discrepancy is in our view a lack of accessibility of development resources and know-how. Smartcard systems are generally of a closed and proprietary nature, carefully guarded by industry through pervasive use of strict non-disclosure agreements (NDAs). Hardly any open and free software exists to pursue smart card research in academia. Perhaps for this reason, industry is doing most of the constructive research in this field while academic research limits itself largely to the investigation and improvement of attack vectors [15,19] or focuses on cryptographic aspects. Other research such as [22,14] that produced results applicable to smartcards was conducted on alternative platforms because with the limited access on available smartcards the required experiments were not possible.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 277–290, 2012.

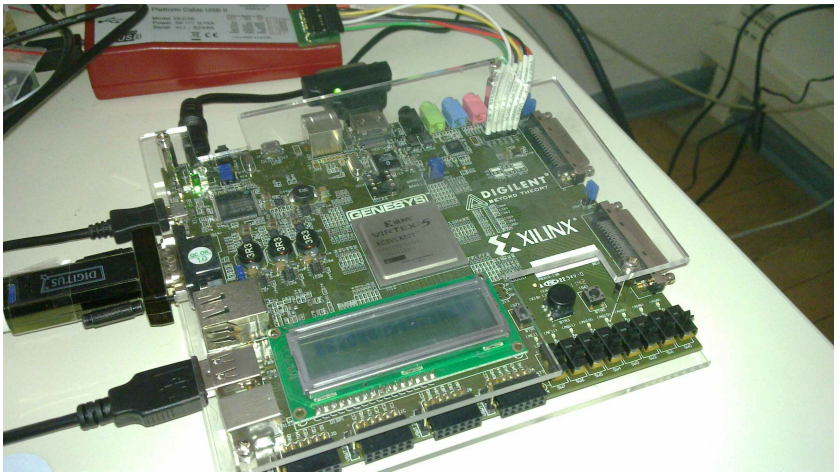


Fig. 1. The hardware platform

This also affects the state of education in the smartcard field. Most courses are introductory and concern themselves only with Java Cards, which allow easy programming but limit the opportunity to understand smartcards at a deeper level. An in-depth treatment of smartcards is hardly ever possible because the material that is necessary for instruction is not available without restrictions.

Contributions: We believe we can make a positive impact on research and education in this field by creating an open research platform. Towards this goal, we report on the following contributions: We designed and built a smart card research platform that is based on free software or software that is free at least for non-commercial use. A *field programmable gate array* (FPGA) serves as a flexible hardware basis (see also Figure 1). With this choice, we trade development flexibility against, for example, the typical smartcard form-factor (a plastic card) and support for hardware security features.

We discuss in this paper how we arrived at the architecture of our platform and its benefits. Its open nature allows to monitor, debug, modify, and extend this platform at every level, even the lowest ones, and at runtime. Hardware changes or additions can be accomplished simply by loading hardware descriptions onto the FPGA. Lastly, the platform can be reproduced with limited funds, and it can be used without having to enter into non-disclosure agreements. In summary, our platform should be accessible for the purpose of research and education.

2 Background

In this section we introduce salient smartcard properties and components for readers who are not already familiar with them, and we outline the properties an "ideal" smartcard research platform should have. Subsequently, we discuss

in greater detail why there is only limited academic research in this field using previous approaches, and the challenges they encountered, as examples.

2.1 Smartcard Characteristics

The main components of a smartcard are its communication interfaces according to ISO/IEC 7816 (based on contact pins) or ISO/IEC 14443 (contactless, based on radio) and the smartcard logic or operating system that needs to comply with ISO/IEC 7816. The main logic is contained in the file-system. Complex smartcard applications can be created through connections between objects, such as keys, and operations on objects with security states. Smartcards usually contain three kinds of memory. The program code of the OS and applications are stored in *read-only memory* (ROM). *Random-access memory* (RAM) is used to execute operations, and non-volatile memory, such as *electrically erasable programmable read-only memory* (EEPROM) contains file-system objects. As an alternative to EEPROM, smartcards may have Flash memory, for example, in the case of the STM ST33F1M [10].

2.2 Requirements

An ideal smartcard research platform would certainly be an actual, programmable, smartcard with debug interfaces such as JTAG. This would enable accurate and practical research on hardware and software aspects of smartcards. We are confident that such devices actual exist in the labs of manufacturers, but access to them is restricted (at the very least by NDAs) as it is certainly possible to reverse engineer or extract valuable intellectual property (IP) from them.

Our goal is to create an open platform for research that conforms as closely as possible to salient smartcard properties. This said, our research focus is on smartcard software. With this perspective in mind not every aspect of smartcards is equally important to our effort. For example, the smartcard form factor is of secondary importance, as long as the platform supports the type of connectivity that is typically expected of smartcards. While hardware security features are of utmost importance to commercial smartcard developments they are of lesser importance to our effort at this time. Perhaps the most important aspect is the hardware abstraction and the operating system. Based on this rationale, we require the following components and features:

1. The platform should provide contact interfaces according to ISO/IEC 7816, and contactless interfaces according to ISO/IEC 14443.
2. The platform should allow complete access to its internals and to debugging facilities.
3. Programming and testing cycles should be quick and simple, that is, without multi-step procedures.
4. The platform should be affordable for research groups with a limited budget.
5. The hardware of the platform should be easy to modify and extend.

Our requirements thus comprise of *features, transparency, usability, cost* and *flexibility* components. Particularly flexibility is an interesting requirement as several industrial and governmental institutions begin to investigate smartcards with new input and output modalities. To the extent possible, a small form factor and hardware security features are desirable, for example, measures that counteract *differential power analysis* (DPA).

2.3 Challenges and Opportunities

Most pervasive systems spur considerable research of some kind or another. It is therefore at first astonishing that smartcard research is limited in academia at the time of writing. We offer two reasons for this situation:

1. Only recently have hardware development boards become available at a cheap price that allow easy assembly of representative sets of components, for example, the much renowned *Arduino* boards or FPGA development boards such as the one we use for our platform. Beforehand, additional expertise was necessary and an effort that is not typically straightforward for researchers interested in working on smartcard systems software.
2. Industry has accumulated a rich expertise in designing smartcard systems yet guards it jealously against outsiders and competitors. If expertise is shared at all, in the form of documentation or development systems, then it is walled by means of stringent NDAs. From an industry perspective their strategy of secrecy is very well motivated. It helps retaining competitive advantages, it cloaks potential, inadvertent and infringing use of competitors' IP, and it increases the effort necessary to attack smartcard systems.

At the same time, the secrecy strategy has downsides. First, limited scrutiny of widely deployed smartcard systems by virtually everyone but resourceful adversaries may obscure the presence of a critical flaw. Once deployed, it may be prohibitively difficult or costly to fix such a flaw, that is, without a complete replacement of the system's deployed components. Second, while the industry has some incentive to stay ahead of the competition, the incentives to push the envelope of what is feasible, or to be innovative, may be limited. Here, secrecy forfeits the opportunity for newcomers to create disruptive technologies as we have seen it happen recently in the smartphone market. Third, secrecy forfeits the creative potential of public academic research. As an example of this creative power we would like to point to advances in cryptography subsequent to its escape from the confines of closed military research. Fourth, industry is solely responsible for the education of its next generation of experts as non-trivial background is rarely taught in university curricula.

2.4 Earlier Approaches

To the best of our knowledge none of the freely accessible operating systems that existed for smartcard devices at one point or another is still actively maintained.

The not free systems offer to little access to serve as a research platform with our requirements.

Contemporary courses in education often use Java Cards [13], which are a de facto industry standard. Oracle, succeeding SUN, provides basic development software for Java Cards without fees. The development environment includes simulation software and debug tools. However, more popular proprietary development tools exist with which the development of complex applications is noticeably easier. Java Cards have clear advantages from an industry point of view. They are a good introduction to the field and they rely on a subset of the Java standard which, at the time of writing, is part of most programming courses. There requirement for special training in smartcard specifics is limited. However, it is not possible to access the lower layers of the card, starting from to JVM as the JVM and Global Platform [17] are the only intended interfaces. Neither is it feasible to add new functionality outside of the VM, modify the OS or to understand the inner workings of the hardware, without close cooperation with a card manufacturer or OS vendor.

The SOSSE project [12] was meant to enable general access to smartcards by providing a freely programmable and changeable OS licensed under the GPL. The situation at the time regarding hardware was similar to the present. Access to security features is only possible under NDAs. The SOSSE project therefore built on Atmel AVR cards which were used for example as Pay-TV access tokens. These cards were available in the typical smartcard ISO/IEC 7810 ID-1 form factor and were user programmable with suitable programmer hardware. The Atmel AVR cards did not support any of the security mechanisms that were state of the art ten years ago and therefore did not offer any protection against sophisticated attacks.

While SOSSE reached a point where it appeared usable, it lacks support for standard cryptographic algorithms and it is restricted to the hardware it was developed for. The integrated algorithms are only exemplary. One useful feature of the project is the ability to simulate the OS on desktop hardware by cross compilation and simulation. SOSSE development stalled in 2003 when the main developer took a position in the smartcard industry and abandoned this hobby [16]. The exact Atmel Pay-TV smartcards that SOSSE used are not in production anymore. While compatible hardware is still available, access is limited to a defined API and it does not support debugging. The possible access extends further than on Javacards but is still to limited.

Some existing devices offer functionality comparable to our stated requirements. The OpenPCD project [4] produced an RFID reader/writer/emulator based on available hardware for which the schematics, layout, and software sources are freely available. A similar project is the Proxmark III [7], which has similar capabilities and an FPGA to assist the radio interface, which provides flexibility and a broader range of application. These devices offer a lot of possibilities but their use is geared towards attack scenarios. They also require specialized knowledge before they become usable tools and their audience are already highly specialized professionals. In summary, the main focus of these

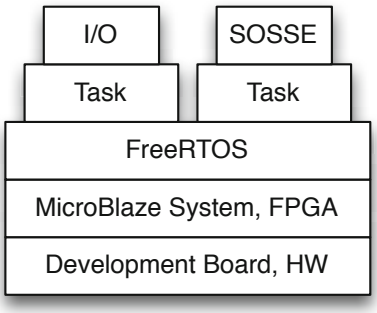


Fig. 2. The platform architecture

projects has been the hardware. The software, though adaptable, does not meet our requirements.

3 Architecture

In this section we describe the components of our platform named FlexCOS. As we have mentioned before, we would like to have the ability to access any part of the platform and being able to modify it as much as possible. This led us to different choices regarding the hardware in comparison to previous projects. The hardware in turn influences our software choices.

We give a schematic overview of our platform in Figure 2. The platform is built on an FPGA configured with a softcore CPU and several hardware IPs that together form a complete system. On top of this we run a small embedded OS which abstracts the underlying hardware. The actual smartcard behavior is implemented by running code that we derived from SOSSE in a task of the OS.

3.1 FPGA

We have chosen an FPGA development board as the basis for our platform. It is equipped with a Xilinx Virtex 5 FPGA, 32 MB of Flash memory and 256 MB of RAM. It has numerous I/O ports, such as Ethernet, USB, HDMI, audio ports, Digilent Pmod connectors, and one serial port. Most of these are not used for our platform. It also contains a small LCD, around 10 LEDs, and a few input buttons. It can be accessed and programmed over USB and with a JTAG debugger. We connect external Flash memory on a Pmod connector.

3.2 Softcore CPU

The use of an FPGA requires the use a softcore CPU. The Xilinx Embedded Development Kit (EDK) includes the Xilinx MicroBlaze [11]. Our system design is centered around this CPU, which is a 32-bit RISC machine based on a Harvard

architecture, designed and optimized for FPGA circuits. The CPU includes an optional MMU and therefore supports virtual memory. This technically allows to run full operating systems such as Linux on it, but we do not make use of this feature yet.

Designing an embedded System with the Xilinx Software based on the MicroBlaze yields a complete system with all required components. The actual parts always depend on the used FPGA and board. In our case this includes DDR2 SDRAM and Flash memory and associated controllers, an interrupt controller, clock generator and timer, as well as two bus systems to connect the instruction and data memory controllers and the other components to the MicroBlaze. Additionally we use a UART for the RS 232 port of the board for communication.

3.3 Operating System

In order to make the programming of the actual functionality easier and to handle the complexity of supporting multiple hardware platforms, we need a hardware abstraction layer. Rather than creating a new abstraction layer from scratch we choose an existing real-time OS, that is, FreeRTOS. This yields a minimalistic system with an interrupt-driven scheduler that supports tasks, queues, message passing, and synchronization mechanisms.

FreeRTOS does not support MMUs and hence no virtual memory, although the ARM Cortex M3 port of FreeRTOS supports the (optional) MPU for this micro controller. Since we do not intend to run multi-threaded applications, which is atypical for smartcards anyways, the lack of virtual memory support is not a limitation. Although, from a security perspective, hardware supported memory protection is a desirable feature for the secure separation of software modules.

3.4 Smartcard Logic

The actual logic and behavior of the smartcard is provided in an abstract layer on top of FreeRTOS. Programatically this distinction in layers is hard to enforce, since the sources of the OS and the smartcard logic are compiled together into a single binary (loadable libraries would be too complicated to be worthwhile on a smartcard system). However, logically we aim to keep the separation between the hardware abstracting code of the real-time OS and the smartcard specific logic. For the latter task we use code from the SOSSE [12] project. Our system runs SOSSE in a FreeRTOS task that represents the main loop. The input and output handling of the platform is currently divided into two tasks triggered by interrupts. The input handling task signals the main loop when data is available. The main loop reads the C-APDU, produces the R-APDU, and signals the output task to send it.

3.5 Extensions

There are certain specifics of our platform that differ from real smartcards. We do only have RAM and Flash available compared to ROM and usually EEPROM. Not having ROM is offset by the FPGA providing settings to automatically load an image from an address in Flash. Not having EEPROM requires us to also use the Flash for non-volatile data such as the file-system. Although, Flash is starting to displace EEPROM in modern smartcards, using the integrated Flash of the FPGA is disadvantageous since it is not writable with fine granularity. Changing a single Bit requires a delete and rewrite of a complete page of memory. On our FPGA this is usually 128KB in size, making these operations uneconomical. A simple solution is to use external Flash memory, which is available commercially as an extension. Such a module has a page size of 256 Bytes, which is more suitable for a smartcard file-system.

Because we do not use something analogous to a SDK, we have to code the hardware interaction ourselves. This means that the receiving and sending routines as well as the memory access routines have to be coded in FreeRTOS and are therefore dependent directly on the FPGA-configuration parameters. We also use Xilinx Libraries for hardware and memory access, that also require a Xilinx FPGA as the target during compilation.

For coding we use a development environment that is commercial software, the SDK from the Xilinx Design Suite, which is publicly available in a royalty-free version.

Currently we do not have a communication interface according to ISO/IEC-7816 or ISO/IEC-14443. Instead we are using a serial connection between the FPGA and a host computer. This works satisfyingly for the moment but requires some adaptations on the host side. For its ease of modification we use the PCSC-lite [6] daemon available for Linux/Unix/MacOS with a self written IFDHandler, that sends APDUs over the serial port. To test the functionality of the platform we use the cyberflex-shell [21] as it allows to enter raw APDUs on the command line and also supports scripting.

4 Analysis

Here we discuss why we have chosen each specific component, what advantages and disadvantages are connected with the choice and what alternatives do exist. When choosing the components of our platform we had to make trade-offs between some of the smartcard properties. Hardware with security features seems not to be available without NDAs and hardware conforming to the size limitations offers only restricted access. We evaluate how well our platform conforms to the requirements of an ideal smartcard research system we stated in section 2.2.

4.1 Hardware

Evaluating the possible choices for a suitable hardware platform, we did consider choosing concrete development boards offered by several vendors targeting

embedded systems. Very early on it became clear that all the processors and microcontrollers containing smartcard-level security mechanisms were not available without NDAs. This is contrary to our goal of creating an open system since we might have been inhibited to publish source code, which is essential to reproduce our results and build upon them. This led us to the conclusion that we could not use hardware with integrated security features.

An alternative is to completely disregard the security features and only focus on the form-factor. This was the approach taken by SOSSE. Yet the only freely programmable smartcards are the ones used mainly for PayTV. They have the described access limits, require a multi-step procedure for programming, and are hard to debug.

If we look at certain chips that are used in smartcards we sometimes see that they are based on a non-security chip that is available on a development board. In the case of the ARM-architecture this is the SecurCore [1] series of chips, SC000, SC100, and SC300 which are based on the ARM Cortex M0 and M3. Equally, the SmartMIPS [9] chips are based on the MIPS32 4KSD design.

A board that is based on such a chip that can be produced in a fitting form factor would be a preferable choice. Nevertheless, acquiring the hardware actually produced in this form factor is still a challenge, reducing our potential benefit of being able to demonstrate our developments on real hardware. Additionally, using such a board would limit us to one fixed architecture and hardware setup, requiring multiple boards to provide support for different architectures or hardware components.

This made us reconsider the use of standard development boards in favor of an FPGA. Overall, this choice offers several advantages. An FPGA enables us to program a chosen architecture onto it, allowing us the flexibility of switching our hardware platform by reconfiguring the FPGA. We can not argue that FPGAs are designed with security in mind [18] nor that an FPGA in a smartcard form factor does exist. However, the FPGA emulates hardware and a hardware description has to be compiled and flashed into it. This description can as well be used to manufacture hardware that behaves identical to the FPGA, possibly in a form factor suitable for a smartcard. This workflow is indicated in Figure 3. A further advantage we would not be able to realize on other hardware is the ability to incorporate new or changed hardware components into our design. The FPGA allows the integration of arbitrary IP-cores enabling us to experiment with hardware changes. Work such as [14] additionally shows, that the FPGA approach enables research, that would otherwise be hard to accomplish.

From an economic point of view FPGAs have become comparable cheap. Many labs are already equipped with such boards from previous projects. Specialized development boards are often only manufactured in small numbers and are therefore expensive compared to generic FPGAs.

One disadvantage lies in the proprietary nature of FPGAs. Proprietary software from the FPGA vendor is required to create and modify a configuration. The functionality to only load a provided configuration onto the device is provided by software that is publicly available without costs. The functionality to

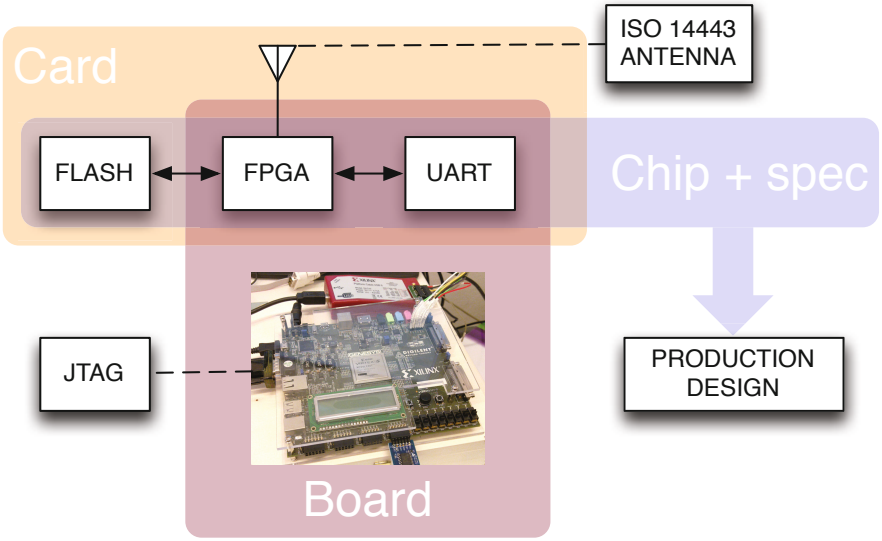


Fig. 3. The FPGA based design

modify the hardware needs an extended license. Since we have chosen a Xilinx based FPGA it is however possible to acquire a royalty free license from Xilinx for academical use through their university program.

4.2 Softcore CPU

As we can see, modern smartcards such as [10] are based on 32 Bit Cores that have MPUs or even MMUs [9]. In architecture they are similar to the MicroBlaze, except for security features such as hiding and masking. The advantage of using the MicroBlaze is its ease of use through its integration into the Xilinx EDK. System-design with this softcore can be accomplished without writing any source code. One disadvantage in using the MicroBlaze consists of it being provided under a non-free license. It is distributed with the Xilinx EDK as a parametric netlist. The HDL-sources can be purchased from Xilinx. This means that the hardware description created by the Software exists only in encrypted form, not allowing any changes to the hardware without a valid license. Also, it may only be distributed in binary form. According to the Xilinx end user license agreement, such binaries may only be used to program Xilinx devices, which makes the platform in binary format dependent on a Xilinx FPGA. In practice the license restriction seems rarely inhibiting. In labs that already have an FPGA a license is often already available since the use of the FPGA without this software is limited. Alternatively, a license for solely academic purposes can be acquired through Xilinx's university program. Still, we are planing to support additional softcore CPUs in the near future, to further the openness of the platform. There are several different free softcore CPUs available [3]. The majority of these are 8 and

16 Bit micro controllers which do resemble some of todays smartcards, but are outdated by todays technical standards. Some firms also offer softcore variants of their chips, for example ARM's Cortex-M1, but they are only available under commercial licenses prohibitively expensive for research and education. A few projects have created LGPL licensed clones of the MicroBlaze, for example [8]. These will be our first targets to support, since the porting effort is minimal.

4.3 Operating System

Although, the codebase of such an OS is relatively small it would have taken us a considerable amount of time to create our own and bring it to a point where it would work stable enough to be used. The choice of free and open OS's for embedded systems is almost unsurveyable. Wikipedia lists more than a hundred projects. After examining most of them we chose FreeRTOS [2] because it offers the best possible combination of advantages.

First, it has broad support for basically all relevant hardware platforms we did consider as a starting point. At the time of writing the official build supports 31 architectures with various toolchains. There are also unofficial ports, supporting even more architectures. Second, the code is manageable, consisting of three main C files and a little architecture dependent code for every port. This architecture dependent code is used for basic initialization and context switching, and therefore has to be partly in assembler. Third, the code is documented extensively, and used in many projects by a large number of people, forming a sizable community of users and developers. The source code is provided under a modified GPL license, so we are able to distribute it in source form.

4.4 Smartcard Logic

Starting to develop a smartcard OS from scratch would also take an unreasonable amount of time, so we decided to take the SOSSE [12] project as a starting point. As described above the state of this project is still far from being complete, but it is usable and allows us to have a working system in a short amount of time. The SOSSE code was adapted directly into a task of the realtime OS since it was designed as a single threaded finite state machine. The hardware it was originally developed for provided a hardware abstraction-layer so that hardly any hardware specific code had to be modified, apart from memory addresses and memory access.

4.5 Evaluation

The platform as described in detail above is currently in an early stage of development, where we have reached an initial working state. Since the SOSSE project had only a narrow amount of functionality and we do not have a usable file-system yet, the supported commands are still limited. The communication is realized over a serial port with a minimal protocol to assure that only complete APDUs are delivered.

We are working to integrate a ISO/IEC 14443 compatible communication interface to replace the serial port.

The integration of new functionality into the our framework is straightforward enabling us to extend it. The platform allows programming and debugging even while running the software on the FPGA. Practically this means that changes to the software made in the SDK can be applied directly by starting a debug-session that flashes the modified binary onto the hardware in just a few seconds.

One of our goals when creating this platform is to make it reproducible. This is achieved by using off the shelf components and publicly providing as much of the source code as possible. The required hardware consists of an FPGA board with a serial port, a serial cable, and an external memory module. We do use a powerful FPGA, a Digilent Genesys with a Virtex 5 Chip, which can be considered over-sized for our needs. We expect smaller and cheaper FPGAs to work as well but have not yet tested them. According to the Xilinx Software our current FPGA configuration uses up to 22% of available resources. It is hard to make comparisons regarding resource use between different FPGA architectures. But judging from the amount of resources needed for the MicroBlaze softcore, the largest part of our configuration, we are confident that this leaves a considerable margin for less powerful FPGAs to be usable. The used Xilinx Software for programming is in large parts freely available (for non-commercial purposes). Only the MicroBlaze softcore CPU requires a license which is needed for the software used to create and change FPGA configurations.

The platform does possess the flexibility to add new hardware or to change the current configuration allowing us to integrate devices such as [20] *Physical Unclonable Functions* (PUFs). We also can add components such as displays or input buttons to study how these extension can influence the user interaction with a smartcard without the requirement to go through a costly design and production process to manufacture samples.

Out of the secondary smartcard characteristics we do not conform in regards to form factor and security features, as this seems challenging without NDAs. The hardware we use is neither designed for security, nor do we have the resources and expertise that an electronic engineering project to design hardware security would require at this time. Our focus and aim currently lies on the software development aspect of smartcards. It is conceivable the we engage in cooperations on hardware security in the future.

The source code of all the components will be made available under the licenses required by the used components. For SOSSE this is GPL Version 2, for FreeRTOS a modified GPL license. The external component in form of the modified IFDHandler under the GPL version 3. We plan to provide the FPGA configuration with the MicroBlaze in binary form. This binary will be specific to the Xilinx development board it was created for. However, it is possible to acquire a free license from Xilinx for academical use through their university program. With such a license a hardware configurations for different FPGAs can be created.

5 Conclusion

We described a platform for smartcard research that allows access to the software as well as the hardware. This enables researchers to modify and extend it, or understand the inner workings of smartcards in a considerable more extensive way as has been possible up to now. All the components except the currently used softcore CPU are free software and we provide them to the public. The hardware can be one of several widespread and affordable FPGA boards, already at hand in many labs. We like to see this platform used in education and as a base for further research into the software components on smartcards.

5.1 Further Research

We are not aiming to provide the same for the hardware security aspects, as this is not our area of expertise, so we leave this aspect of further research explicitly open.

The platform does still have certain shortcomings we are planning to address in the near future. One of the most interesting issues is the communication interface. We have to integrate a contact interface according to ISO/IEC 7816 and/or a contactless interface according to ISO/IEC 14443. Although the contact interface might be easier to realize technically, we rather create a cheap contactless interface that is simple to reproduce. The radio interface offers higher communication speeds and is used in most high volume produced cards nowadays. Trough the integration of such a communication interface the platform can also be used for protocol analysis or testing of software that uses smartcards. This will increase its utility and expand the group of potential users.

As mentioned above memory protection features are a desirable property to implement separation. The integration of such hardware enables experimentations with untrusted code in trusted environments.

To offer the complete platform under open licenses we want to extend support to include free softcore CPUs. Our prime candidates are the OpenRISC or1k2 [5] and the SecretBlaze [8]. These will enable us to make the full stack of components available in source form.

The main improvements we are focusing on next are the file-system, as it encapsulates the majority of the smartcard security model, and a basic cryptographic library, which is needed to realize any useful protocols.

Possible improvements, we currently are not pursuing are a JVM integration that would allow us to run Java Card applets.

Acknowledgments. This work is funded through a grant by the Bundesdruckerei GmbH.

References

1. ARM SecurCore Processors,
`http://www.arm.com/products/processors/securcore/index.php`

2. The FreeRTOS Project, http://www.freertos.org/
3. OpenCores, http://opencores.org/
4. OpenPCD, http://www.openpcd.org/
5. OpenRISC OR1k2, http://openrisc.net/
6. PCSC lite project, Middleware to access a smart card using SCard API (PC/SC), http://pcsclite.alioth.debian.org/
7. Proxmark III, http://code.google.com/p/proxmark3/
8. SecretBlaze, http://www2.lirmm.fr/~barthe/index.php/page/SecretBlaze.html
9. SmartMIPS, http://www.mips.com/products/architectures/smartmips-ase/
10. STMicroelectronics ST33F1M Smartcard MCU, http://www.st.com/internet/mcu/product/215291.jsp
11. Xilinx MicroBlaze, http://www.xilinx.com/tools/microblaze.htm
12. Brüstle, M.: Simple Operation System for Smartcard Education (April 2003), http://www.mbsks.franken.de/sosse/
13. Oracle Corp. Java Card, http://www.oracle.com/technetwork/java/javacard/
14. Ege, B., Kavun, E.B., Yalçın, T.: Memory Encryption for Smart Cards. In: Prouff, E. (ed.) CARDIS 2011. LNCS, vol. 7079, pp. 199–216. Springer, Heidelberg (2011)
15. Garcia, F.D., van Rossum, P., Verdult, R., Schreur, R.W.: Dismantling SecureMemory, CryptoMemory and CryptoRF. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, pp. 250–259. ACM, New York (2010)
16. Brüstle, M.: (Executive Masktech Germany. Personal Communicacion (July 2011)
17. Inc. GlobalPlatform. GlobalPlatform, http://www.globalplatform.org/
18. Moradi, A., Barenghi, A., Kasper, T., Paar, C.: On the vulnerability of FPGA bitstream encryption against power analysis attacks: extracting keys from xilinx Virtex-II FPGAs. In: Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, pp. 111–124. ACM, New York (2011)
19. Oswald, D., Paar, C.: Breaking Mifare DESFire MF3ICD40: Power Analysis and Templates in the Real World. In: Preneel, B., Takagi, T. (eds.) CHES 2011. LNCS, vol. 6917, pp. 207–222. Springer, Heidelberg (2011)
20. Pappu, S.R.: Physical One-Way Functions. PhD thesis, Massachusetts Institute of Technology (2001)
21. Plötz, H.: Cyberflex-Shell, https://github.com/henryk/cyberflex-shell
22. Rahmati, A., Salajegheh, M., Holcomb, D., Sorber, J., Burleson, W.P., Fu, K.: TARDIS: Time and remanence decay in SRAM to implement secure protocols on embedded devices without clocks. In: Proceedings of the 21st USENIX Security Symposium, Bellevue, WA (August 2012)

Towards Formalizing a Reputation System for Cheating Detection in Peer-to-Peer-Based Massively Multiplayer Online Games

Willy Susilo[1,⋆], Yang-Wai Chow[2], and Rungrat Wiangsripanawan[3,⋆⋆]

[1] Centre for Computer and Information Security Research
[2] Centre for Multimedia and Information Processing
School of Computer Science and Software Engineering
University of Wollongong, Australia
{wsusilo,caseyc}@uow.edu.au
[3] Department of Computer Science, Faculty of Science
King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
kwrungra@kmitl.ac.th

Abstract. The rapidly growing popularity of Massively Multiplayer Online Games (MMOGs) has given rise to an increase in the number of players world wide. MMOGs enable many players interact together through a shared sense of presence created by the game. The Peer-to-Peer (P2P) network topology overcomes communication bottleneck problems associated with centralized client/server systems. However, many architectures are proposed in an ad hoc manner and enhancing the security of such systems is an elusive research problem. In this paper, we address this important issue by making the following contributions. Firstly, we formalize the notion of P2P-based MMOGs and demonstrate that existing P2P-based MMOG architectures can be unified using our model. To our knowledge, this is the first time that this has been done in the literature. Secondly, we use our model to develop a real-time cheating detection mechanism to identify cheating players, which can be used to expose several MMOG cheating strategies. Finally, we propose a new reputation based system for P2P-based MMOGs to enhance the cheating detection process.

1 Introduction

Massively Multiplayer Online Games (MMOGs) are online games which provide networked virtual environments where many players, typically ranging into the thousands, can interact with other players through a shared sense of presence created by the game. In MMOGs, players might physically be located all over the globe, but should be able to comfortably interact within the shared environment. The popularity and success of MMOGs has led to an increase in the number of

⋆ This work is supported by ARC Future Fellowship FT0991397.
⋆⋆ This work was done when the author visited University of Wollongong, Australia.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 291–304, 2012.

users world wide. Consequently, the scalability of MMOG network architectures has become a key challenge that has to be addressed.

To date, many successful MMOGs are predominantly based on the Client/Server (C/S) network topology [4, 14]. In C/S systems, the centralized servers create a bottleneck as all communication must pass through the servers. This gives rise to a single point of failure, and expensive game servers have to be used to handle the large computational requirements of the system [8]. Furthermore, this centralized approach results in a huge amount of network traffic at the server-side, increases the communication latency between clients and inhibits the scalability of the system.

The Peer-to-Peer (P2P) network topology on the other hand overcomes the communication bottleneck problems associated with centralized servers by distributing computational load among the peers [8]. This allows for greater scalability, avoids the cost of expensive servers, and potentially reduces latency between interacting peers. As such, over the years researchers have proposed a variety of scalable P2P-based network architectures for MMOGs (such as [11, 16, 24]).

2 Background

2.1 P2P-Based MMOG Network Architectures

We adopt the terminology and definitions used in [4,17] to represent the different types of P2P-based MMOG architectures. For a detailed survey, please refer to [4, 17].

ALM Based Protocols
In the Application Layer Multicast (ALM) approach, game events and messages are distributed using standard ALM techniques. In many implementations, the virtual game world is partitioned into subspaces or spatial regions. Each region is represented by a dedicated multicast group, and events within that region are sent to all relevant players in that region. Players only need to be informed of events happening within a certain range in the virtual environment. This range is known as the Area of Interest (AOI) and was a concept first introduced by Macedonia et al. [20]. In many cases, a player's AOI is fully inside a single region. However, if a player's AOI intersects the border between regions, he/she also has to subscribe to the other region's multicast group. Examples of the ALM based protocol can be found in [6, 7, 13, 16, 21, 22].

Supernode Based Protocols
Similar to the ALM approach, the virtual game world is also divided into spatial regions. In some implementations, the region size is fixed [24], while in others, region sizes change dynamically based on player density in order to balance computational load [5]. For each region, a supernode, or superpeer, is selected and assigned as the coordinator for that region, effectively acting like a region server. The supernode is responsible for receiving all game event messages within the region and disseminating these to all players that are subscribed to that region.

Supernode based protocols are used in [3,12,14,24]. These supernode models are all based on the assumption that there is a way to choose a trustworthy node to act as the supernode for each region.

Mutual Notification Based Protocols
This approach does not involve explicitly dividing the virtual game world into spatial regions. Instead, players send messages directly to other players within their AOI. Thus, message and event propagation delays are minimized. In mutual notification based protocols, players must be aware of all other players within their AOI. As such, players must depend on their neighbors for information regarding other players who have recently moved into their AOI. This protocol is used in [10, 11], where each player computes a Voronoi diagram based on all known neighbors. Whenever a player changes location, all neighbors must be notified so that they can updated their own local Voronoi diagrams. Neighbors are added to, or removed from, a player's notification list based on changes in Voronoi diagram information. Even though the virtual game world is not explicitly divided into spatial regions, in some sense the Voronoi diagrams still dynamically form non-uniform regions for mutual notification.

3 Formal Model of P2P-Based MMOG

3.1 High Level Description

The system described in this section will be a generic system that can be applied to the different P2P-based MMOG architectures previously discussed. While MMOG architectures are extremely complex systems that are made up of an amalgamation of diverse factors, many of these factors do not directly relate to cheating detection. As such, we will only focus on factors that will aid us in the task of detecting cheating peers. For example, P2P-based MMOGs may maintain a login server, whose main tasks include checking player subscriptions before allowing players to join the game, assigning a player to a region upon joining and providing initial information for the player to link with his/her peers. We will not consider such factors in our model, as our focus is on the P2P architecture that underlies the running of the in-game environment.

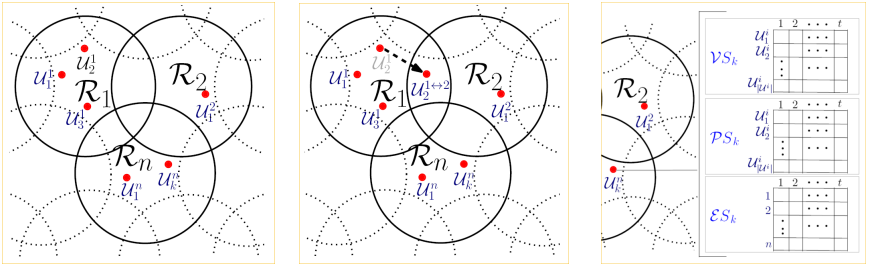
Typical P2P-based MMOG architectures are composed of a number of regions, whether fixed sized, dynamically changing with respect to player density or determined based on player AOIs. In many cases, a player's AOI is fully contained within a single region. At any given time, a player mainly resides in one of the defined regions. In which case, we say that the player is 'subscribed' to that region. If a player's AOI intersects the border of a neighboring region, then he/she has to also subscribe to the other region. On the other hand, the player 'unsubscribes' from a certain region if that region is no longer relevant to the player. Each region is identified with an ID, which may be implemented by simply using a collision resistant hash function.

3.2 Formal Definition and Model

Setup

A P2P-based MMOG system comprises of n regions denoted as $\mathcal{R} = \{R_1, \cdots, R_n\}$. Each region, R_i, is identified by an identity ID_{R_i}. When a user $\mathcal{U}_j$ resides in a region R_i, we denote it as $\mathcal{U}_j^i$. Each region R_i contains m users at some stage, and therefore we denote it as $\mathcal{U}^i = \{\mathcal{U}_1^i, \mathcal{U}_2^i, \cdots, \mathcal{U}_m^i\}$. The total number of users in $\mathcal{U}^i$ is denoted as $|\mathcal{U}^i|$, which is equal to m in the above case. Each user in $\mathcal{U}^i$ is said to have 'subscribed' to R_i. This is illustrated in Fig. 1(a). When a user $\mathcal{U}_k^i \subset \mathcal{U}^i$ moves and interacts with R_j, we denote it as $\mathcal{U}_k^{i \leftrightarrow j}$, which implies[1] $\mathcal{U}^j_{|\mathcal{U}^j|+1} := \mathcal{U}_k^i$ and $\mathcal{U}^j := \mathcal{U}^j \cup \left\{\mathcal{U}^j_{|\mathcal{U}^j|+1}\right\}$. We note that this also means $|\mathcal{U}^j| := |\mathcal{U}^j| + 1$, since $\mathcal{U}_k^i$ has subscribed to R_j. When $\mathcal{U}_k^{i \leftrightarrow j_1 \leftrightarrow \cdots \leftrightarrow j_h}$, this implies that for each $\alpha \in \{j_1, \cdots, j_h\}$, $\mathcal{U}^\alpha_{|\mathcal{U}_\alpha|+1} := \mathcal{U}_k^i$ and $\mathcal{U}^\alpha := \mathcal{U}^\alpha \cup \left\{\mathcal{U}^\alpha_{|\mathcal{U}_\alpha|+1}\right\}$. This reflects the situation in which a player's position overlaps the borders of other regions. Fig. 1(b) shows a depiction of this. After the completion of $\mathcal{U}_k^{i \leftrightarrow j}$, meaning that the previous region is no longer relevant, $\mathcal{U}^i := \mathcal{U}^i \setminus \{\mathcal{U}_k^i\}$ must occur, and consequently $|\mathcal{U}^i| := |\mathcal{U}^i| - 1$. This represents the situation where $\mathcal{U}_k^i$ leaves R_i, and so $\mathcal{U}_k^i$ must 'unsubscribe' from R_i.

Let δ_{min} denote the minimum number of users required for a region to be formed. Where $\delta_{min} := 0$, means that there is *no* restriction on the minimum number of users.



(a) Users in the Regions (b) Movement of $\mathcal{U}_2^1$ to $\mathcal{R}_2$ (c) Data Storage of each peer

Fig. 1. Region, Nodes and Data Storage

Data Storage

In order to maintain the state of the game in P2P MMOGs, the peers need to store information about the state of the virtual game environment (e.g. Non-Player Characters (NPCs)), as well as the states of the other peers including his/her own. For supernode architectures, most of this information is stored in

[1] This means that the last user in the region R_j, which is $\mathcal{U}^j_{|\mathcal{U}^j|+1}$, is set to be the new incoming user $\mathcal{U}_k^i$, and the size of the set $|\mathcal{U}^j|$ is increased by one.

a single peer for each region. Three different lists need to be maintained by $\mathcal{U}_k^i$ to record information about the environment and the other peers in that region. These are to be implemented as queues. Let:

- $\mathcal{VS}_k := \{\mathcal{VS}_k^1, \cdots, \mathcal{VS}_k^t\}$ be the list of 'virtual player states'. These states contain the data of all player in the region. Player data might include Health Points (HP), experience points, level, money, items, attributes, etc.
- $\mathcal{PS}_k := \{\mathcal{PS}_k^1, \cdots, \mathcal{PS}_k^t\}$ be the list of 'physical states'. This records the players real world information such as connection speed, average message transmission time, latency, etc.
- $\mathcal{ES}_k := \{\mathcal{ES}_k^1, \cdots, \mathcal{ES}_k^t\}$ be the list of 'virtual environment states'. This is used to store information about non-player entities in the game environment, for example the state of NPCs.

The number of states maintained in the queue is represented by t. This means that if t is set to 10, the last ten states will be stored. As mentioned, this information is stored by $\mathcal{U}_k^i$, which might be a single peer in the case of a supernode architecture, or $|\mathcal{U}^i|$ peers otherwise. Since $\mathcal{VS}_k$ and $\mathcal{PS}_k$ encompasses the data of all players in the region, both virtual and physical, this is denoted as

$$\left(\mathcal{VS}_k^l, \mathcal{PS}_k^l\right) := \left(\{\mathcal{VS}_k^1, \cdots, \mathcal{VS}_k^{|\mathcal{U}^i|}\}, \{\mathcal{PS}_k^1, \cdots, \mathcal{PS}_k^{|\mathcal{U}^i|}\}\right)$$

we abuse the notation (without any confusion) as $\mathcal{VS}_k^l[j]$, where $l = 1, \cdots, t$ and $j = 1, \cdots, |\mathcal{U}^i|$ to denote

$$\mathcal{VS}_k^l[j] := \mathcal{VS}_k^j$$

and $\mathcal{PS}_k^l[j]$, where $l = 1, \cdots, t$ and $j = 1, \cdots, |\mathcal{U}^i|$ to denote

$$\mathcal{PS}_k^l[j] := \mathcal{PS}_k^j$$

respectively.

Additionally, $\mathcal{U}_k^i$ needs to maintain the state of the virtual environment $\mathcal{ES}_k := \{\mathcal{ES}_k^1, \cdots, \mathcal{ES}_k^t\}$. For a region that has n non-player entities

$$\mathcal{ES}_k^l := \left(\mathcal{ES}_k^1, \cdots, \mathcal{ES}_k^n\right)$$

Therefore, in total user $\mathcal{U}_k^i$ in region R_i needs to store

$$t(2|\mathcal{U}^i| + n)$$

information to record the game environment states, as well as the virtual and physical states of all players in the region. Fig. 1(c) shows the information that is stored by the peers, or superpeers. Note that to deter cheating, in certain P2P MMOG implementations the user is not allowed to store his/her own state [19].

Communication

Let δ_{send} denote the set of peers that each user needs to report its states to for every single update cycle. For $\mathcal{U}_k^i$, δ_{send} is defined as $\delta_{send} := \{1, \cdots, |\mathcal{U}^i|\} \neq$

$\{\mathcal{U}_k^i\}$ for R_i. At times the choice of the peers may be defined by proximity gathered from the physical states of other peers $\mathcal{P}S_k$. A special case happens when $\delta_{send} := \{1\}$, since each user needs to report its states to a designated user, $\mathfrak{R}_i$, in R_i. This designated user is often known as the supernode, which is selected using a selection scheme from among the users in R_i.

3.3 Instantiating the Model

Here, we show how our formal model can be instantiated and applied to the existing types of P2P-based MMOG architectures that were described in section 2.1.

ALM Based Protocols

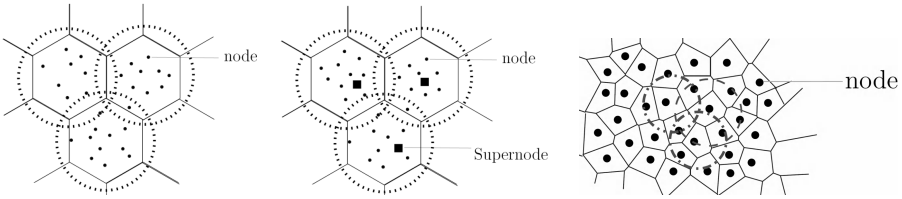
In this architecture, the game world is typically divided into subspaces, and hence will be represented as a collection of R_i's in our model. A collection of players U^i reside in R_i and maintain their respective Area of Interests (AOIs). The way the user subscribes and unsubscribes to a region, based on their AOIs, is as per our model. Typical MMOGs divided the game world into square or hexagon based subspaces. Hence, Fig. 2(a) depicts how R_i in our model can be applied to hexagon subspaces. Similarly, our model can easily be applied to square based subspaces.

Supernode Based Protocols

Supernode based protocols are similar to ALM based protocols in that they are region based. The difference being the existence of a responsible node, called the supernode in each subspace. This follows our model where $\delta_{send} = 1$, and hence, $\mathfrak{R}$ is the supernode. Refer to Fig. 2(b).

Mutual Notification Based Protocols

Unlike the ALM and supernode based protocols, mutual notification based protocols do not explicitly divide the game world into rigid subspaces. Each player interacts with other peers in the system, when their proximities are closed to each other. Since they compute proximities using some method, for example by constructing a Voronoi diagram, these Voronoi regions can be clustered into the R_i regions represented in our model. Peers within R_i indicate the neighbors in which a node directly communicates with. This is illustrated in Fig. 2(c), the circles are examples of how R_i would be formed around the Voronoi regions (note that only a few are shown to avoid over cluttering the figure). The difference between mutual notification based protocols as compared to the previous two protocols, is that this approach is based on dynamically changing regions, which are non-uniform. Hence, in Fig. 2(c) the circle sizes are non-uniform. There must be a minimum number of users required in order to define a region, namely δ_{min}.



(a) ALM based Protocols (b) Supernode based Protocols (c) Mutual Notification based Protocols

Fig. 2. Instantiations of Our Model

4 Cheating Detection

4.1 Detecting Suspicious Behavior

Before defining the cheating action, let

$$\beta \in \{1, \cdots t-1\}, \gamma_{\mathcal{ES}} := \mathtt{Comp}_{\mathcal{ES}}(\mathcal{ES}_k^l, \mathcal{ES}_k^{l+\beta})$$

$$\beta \in \{1, \cdots t-1\}, j \in \{1, \cdots |\mathcal{U}^i|, j \neq k\}, \gamma_{\mathcal{VS}} := \mathtt{Comp}_{\mathcal{VS}}(\mathcal{VS}_k^l[j], \mathcal{VS}_k^{l+\beta}[j])$$

$$\beta \in \{1, \cdots t-1\}, j \in \{1, \cdots |\mathcal{U}^i|, j \neq k\}, \gamma_{\mathcal{PS}} := \mathtt{Comp}_{\mathcal{PS}}(\mathcal{PS}_k^l[j], \mathcal{PS}_k^{l+\beta}[j])$$

where $\mathtt{Comp}_{\mathcal{ES}}, \mathtt{Comp}_{\mathcal{VS}}, \mathtt{Comp}_{\mathcal{PS}}$ define the comparison functions for virtual environment states, virtual player states and physical states, respectively. Depending on the nature of the game, these functions could be as simple as a substraction function, an XOR operation, or something more complex. To reduce a peer's computational load, these functions do not have to be executed every single cycle. Instead, they can be executed sporadically at random intervals, and only increasing the number of executions when a potentially suspicious player is detected.

These functions are used to identify in-game cheating behavior. The cheating action that can be identified by $\mathtt{Comp}_{\mathcal{ES}}$ are those where the cheater tries to propagate false game environment states, for example, killing a strong NPC in a single blow, or falsifying the type or amount of an item in the environment which is not currently owned by any of the peers. $\mathtt{Comp}_{\mathcal{VS}}$ is used to identify whether a cheater tries to maliciously modify his/her own player status, for instance, inappropriately increasing his/her HP, moving through walls or moving at impossible speeds, duplicating an item that he/she owns, etc. $\mathtt{Comp}_{\mathcal{PS}}$ on the other hand is used for detecting network cheats like trying to delaying event propagation to other peers, or changing an update messages' timestamp.

We define a function $\mathtt{CheatDetect}(\mathsf{ID})$ that invokes the following:

- A user $\mathcal{U}_k^i$ in $\mathcal{R}_i$ suspects that there is a user, $\mathcal{U}_\ell^i$, in $\mathcal{R}_i$ that is cheating.
- $\mathcal{U}_k^i$ will multicast ID_ℓ to all other peers in $\mathcal{R}_i$, in order for them to check and verify this suspicion.

- All other peers in $\mathcal{R}_i$ will check and determine for themselves whether or not the user with ID_ℓ is cheating and multicast the same ID to all peers if this is found to be true. If the same ID is received more than once within a certain timeframe, it will be ignored to avoid network congestion due to message flooding.
- If a certain number of 'votes' given by the peers is obtained, a consensus is reached and the suspected user will be marked as a cheater. If no consensus is reached, the data will be ignored, and the situation will return to the status quo.

The function $\texttt{CheatDetect}(\mathsf{ID})$ accepts ID of the suspected user as its input, and it outputs either $\top$ or $\bot$, to indicate whether or not ID^i_ℓ is a cheater. Note that the voting system can be enhanced by using a reputation system. In other words, a vote from a peer with a higher reputation will have a greater weight in the overall decision. This will be elaborated in section 5, which describes our reputation system.

Specifically, a user $\mathcal{U}^i_k$ may suspect that $\mathcal{U}^i_\ell$ is cheating if $\gamma_{\mathcal{ES}} > \tilde{t}$ and/or $\gamma_{\mathcal{VS}} > \tilde{t}$ and/or $\gamma_{\mathcal{PS}} > \tilde{t}$, for a defined threshold $\tilde{t}$. When this condition occurs, $\mathcal{U}^i_k$ multicasts ID_ℓ to $\mathcal{U}^i_m \in \mathcal{U}^i$, where $m = \{1, \cdots, |\mathcal{U}^i|\}$, $m \neq \{k, \ell\}$. Upon receiving ID_ℓ, each $\mathcal{U}^i_m \in \mathcal{U}^i$ will check $\mathcal{ES}^j_m$ and $(\mathcal{VS}^l_m, \mathcal{PS}^l_m)$

$$\beta \in \{1, \cdots t-1\}, \gamma_{\mathcal{ES}} := \texttt{Comp}_{\mathcal{ES}}(\mathcal{ES}^j_m, \mathcal{ES}^{j+\beta}_m)$$

$$\beta \in \{1, \cdots t-1\}, \gamma_{\mathcal{VS}} := \texttt{Comp}_{\mathcal{VS}}(\mathcal{VS}^l_m[\ell], \mathcal{VS}^{l+\beta}_m[\ell])$$

$$\beta \in \{1, \cdots t-1\}, \gamma_{\mathcal{PS}} := \texttt{Comp}_{\mathcal{PS}}(\mathcal{PS}^l_m[\ell], \mathcal{PS}^{l+\beta}_m[\ell])$$

and subsequently, if any of these comparisons indicate that U^i_ℓ has interfered with the update messages, tampered with the data or performed a suspicious action, ID_ℓ will be multicasted to $\mathcal{U}^i_m \in \mathcal{U}^i$, where $m = \{1, \cdots, |\mathcal{U}^i|\}$, $m \neq \{k, \ell\}$. Let $\bar{t}$ denote the threshold required to judge whether or not a user is a cheater. The value of $\bar{t}$ must be based on the total number of users in the region, i.e. $|U^i|$. If the number of ID_ℓ received is greater than $\bar{t}$, U^i_ℓ is identified as a cheater, and the output of $\texttt{CheatDetect}(\mathsf{ID}^i_k)$ will be $\top$. Otherwise, if when the number of ID_ℓ received given a number of cycles is less than $\bar{t}$, then its output will be $\bot$. Depending on the design of the game, a cheater may immediately be kicked from the game, and have his/her account suspended or banned from joining the game in future.

In a supernode architecture, $\mathcal{U}^i_k$ is a trusted peer and is responsible for identifying cheaters without the help of other peers. In this case, $\mathcal{ES}$, $\mathcal{VS}$ and $\mathcal{PS}$ are stored in the supernode. Thus, the supernode itself will run $\texttt{Comp}_{\mathcal{ES}}, \texttt{Comp}_{\mathcal{VS}}$, $\texttt{Comp}_{\mathcal{PS}}$, and collect this information over a number of cycles before determining the output of $\texttt{CheatDetect}(\mathsf{ID})$.

4.2 Cheating Techniques in P2P-Based MMOG

Here, we elaborate several cheating strategies that can be launched by malicious players. We limit our discussion to cheating mechanisms that are related to P2P

MMOG architectures. In addition, our method deals with in-game type cheating techniques. Therefore, we do not examine cheating methods like login cheats or system administrator abuse, as they are not relevant to our discussion. We adapt the terminology of cheating strategies from [25].

Cheating by Exploiting Misplaced Trust
This is a common MMOG cheating mechanism that involves tampering with game code, configuration data, or both, and hence, requires reverse engineering on the client/peer's side. The malicious user, the cheater, can then modify the game client data to whatever value he/she wants. Alternatively, the cheater can also modify the game client in order to alter sensitive game states on the fly. This type of cheating can be detected in our architecture by observing past and present data, which may be stored in a supernode or shared among multiple peers. Specifically, if the cheater attempts to modify the current state of the game, the system can identify this suspicious behavior by comparing this with t previous states using the set of `Comp` functions. This detects cheats like increasing a player's attributes, "speed hacks", "duping items", etc.

Cheating by Modifying Client Infrastructure
The cheater can modify the client infrastructure such as device drivers in his/her operating system. By doing this, for instance, the cheater can make walls transparent (this is known as "wall hack"). In previous work by Laurens et al. [18], they detected "wall hacks" by incorporating the concept of a *trace*. Essentially, the *virtual states* of the each player has to be observed to determine precisely what the player is looking at. The frequency of illegal traces can identify the case where a player keeps looking at objects that the player cannot actually see. Alternatively, suspicious behavior can be determined if a player continually 'stares at a wall', because this is invisible to him/her. Our detection mechanism can handle this type of cheats by storing and observing the frequency of suspicious behavior over a number of cycles.

Timing Cheating
In this type of cheating mechanism, the cheating player choose to delay his/her own move until he/she knows all the opponents' moves, and hence, gaining a huge advantage. This type of cheating strategy can be detected by using our cheating detection mechanism as information about the *physical states* for each player is recorded. By observing the physical states of each peer, our system can determine artificially induced delays or when message timestamps are modified.

Cheating by Exploiting Lack of Secrecy
This type of cheating strategy is performed by illegally accessing game data (or states). This situation can arise in our model if the cheater can somehow obtain the contents of the queues. In general, combating this is straightforward as the state information for each player can be encrypted with a symmetric algorithm, such as AES. Assuming the security of the algorithm is hard (which is the case for the state-of-the-art AES algorithm), this cheating strategy will be rendered ineffective.

5 Adding a Reputation System to P2P-Based MMOGs

We employ a reputation system that is inspired by EigenTrust [15], X^2Rep [2] and X^{2BT}Rep [26] which have been designed for use in P2P networks. Nevertheless, we should stress that the reputation systems proposed in P2P networks cannot directly be used in P2P-based MMOGs. This is because in many large-scale cutting-edge MMOGs, the game itself requires tremendous computational resources to run and all this computation has to be performed in real-time. Any delays, due to network latency or processing load, can severely impact the players' in-game experience. One of the main principles that we employ in the development of our reputation system, is that in MMOGs the main purpose of each individual peer is to protect the player himself/herself, as opposed to trying to protect the entire system.

5.1 High Level Idea

Each user is equipped with a list of reputations of all peers in the region. The user will not store his/her own reputation. When a new user joins a region R_i, the user is given a default reputation Δ. Note that Δ cannot be zero, since a reputation of zero will prevent the user from joining the MMOG game in the first place (this is known as "cold start" in P2P-based reputation system). Upon joining R_i, the user contacts the peers in R_i to obtain the reputation of other peers. When used in cheating detection, a peer's voting weight can be adjusted based on the value of the peer's reputation. Hence, a user with higher reputation will contribute more weight towards determining whether or not another peer is a cheater. In a similar manner, once a user moves into another region, the user needs to contact the peers in that region to acquire the reputation values of the other peers. The user's reputation in that new region will be calculated based on the reputation given by the peers in the region that he/she just left.

This can easily be adapted to a supernode system, where the reputation list would be computed and stored at the supernode. When a user move to a different region, the new region's supernode needs to get the user's reputation from the previous region's supernode. If a supernode changes region, a new supernode is selected and the list is transferred to the new supernode.

5.2 System Design

Each user $\mathcal{U}_k^i$ in R_i is equipped with a list of reputations:

$$\mathsf{Rep}_k^i := \forall_{\ell=1,\cdots,|\mathcal{U}^i|,\ell\neq k}\{\mathsf{Rep}_\ell^i\}$$

We use Rep_ℓ^i to denote the reputation of user $\mathcal{U}_\ell^i$ who resides in region R^i. The list is kept by user $\mathcal{U}_k^i$ to represent user $\mathcal{U}_k^i$'s view on the other peers.

User Joining a Region
When user $\mathcal{U}_k$ enters region R_i (hence, $\mathcal{U}_k^i$), $\mathcal{U}_k^i$ queries δ_{send} peers to acquire the reputation of all other peers in order to fill Rep_k^i. When there is more than one

response received, Rep_k^i is filled with the average of the responses. If there exists $\forall_{\ell=1,\cdots,|\mathcal{U}^i|,\ell\neq k}\{\mathsf{Rep}_\ell^i\} = \emptyset$, then the peer needs to query other peers in R_i. At the end of this process, $\mathcal{U}_k^i$ acquires a complete Rep_k^i. Other peers $\mathcal{U}_j^i \in \mathcal{U}^i$ will assign to $\mathcal{U}_k^i$ with a reputation of Δ.

Moving to Another Region
When $\mathcal{U}_k^{i\leftrightarrow j}, \mathcal{U}_k^i$ queries δ_{send} peers in R_j to acquire all other peer reputations and gather Rep_k^j. Additionally, since $\mathcal{U}_k^i$ has now moved to R_j (hence, $\mathcal{U}_k^j$), the peers in R_j will have to update their records to contain $\mathcal{U}_k^j$'s reputation. However, since $\mathcal{U}_k^j$ does not store his/her own reputation, the peers in R_j will have to acquire $\mathcal{U}_k^j$'s reputation from δ_{send} users in R_i. Similarly, the same process is done when $\mathcal{U}_k^{i\leftrightarrow j_1\cdots j_h}$.

Using the Reputation
The reputation information is used to enhance the quality of the votes given by the peers. When $\mathcal{U}_k^i$ suspects that $\mathcal{U}_d^i$ is cheating, $\mathcal{U}_k^i$ invokes $\texttt{CheatDetect}(\mathsf{ID}_d)$. Upon receiving the ID-s from $\mathcal{U}_\ell^i$, where $\mathcal{U}_\ell^i \in \mathcal{U}^i, \ell \neq \{d,k\}$ computes

$$Res = \sum_{\forall \ell \in \{1,\cdots,|\mathcal{U}^i|\},\ell\neq\{d,k\}} \mathsf{Rep}_\ell^i \times \texttt{IsID}(\mathsf{ID_d}, \mathsf{ID}_\ell).$$

The function $\texttt{IsID}(\mathsf{ID}_d, \mathsf{ID}_\ell)$ will return 1, if ID_d has been returned by user ID_ℓ, or 0 otherwise. If $Res > \bar{t}$, for a threshold $\bar{t}$ as defined earlier in section 4.1, then the user $\mathcal{U}_d^i$ will be marked as a cheater. We call this system a credibility algorithm in P2P-based MMOGs. When a user is identified as a cheater, then his/her reputation is marked as 0.

Reputation Update
If a user $\mathcal{U}_\ell^i$ submits ID_d during a cheating detection phase, and ID_d is eventually marked as a cheater (which means, that ID_d's reputation is marked as 0), then $\mathcal{U}_\ell^i$'s reputation that is stored on the other peers should be updated as $\mathsf{Rep}_\ell^i := \mathsf{Rep}_\ell^i + \xi$. The value ξ is used to increase the reputation of $\mathcal{U}_\ell^i$, because the user correctly identified a cheater $\mathcal{U}_\ell^i$ should now be seen as a more trustworthy peer. A typical value that is used during implementation is 0.05, which is similar to the reputation system used in BitTorrent (X^2BT-Rep [26]).

Conversely, if a user $\mathcal{U}_\ell^i$ submits ID_d during a cheating detection phase, but ID_d is eventually declared to be a non-cheater (i.e. the threshold $\bar{t}$ criteria is not met), then $\mathsf{Rep}_\ell^i := \mathsf{Rep}_\ell^i - c\xi$, where $c \in \{2,\cdots\}$. This essentially means that if a user votes wrongly, then the 'penalty' given is linear to ξ. In contrast, if a correct vote is cast, then the reputation is increased by ξ. This is because a user will normally only vote incorrectly if his/her data diverges significantly from the norm, which indicates that the user is less trustworthy, possibly indicating that he/she has tampered with the system.

While it is conceivable that a player could hack his/her system to avoid submitting votes entirely, thereby never having other peers increase or decrease his/her reputation, this in no way benefits the player as there is nothing that the player gains from withholding votes. Moreover, in a MMOG system which

hosts thousands of players, the chances of having many players hacking their system to withhold votes for no apparent benefit, is extremely remote to have any significant impact on the reputation system. However, it is possible that a player, or a group of players, might be able to reverse engineer their system to maliciously vote against other players in order to kick them out of the game. This is discussed in the section below.

5.3 Security Considerations

A number of security issues that occur in P2P-based MMOG systems are discussed here. In particular, we focus our discussion on security issues faced by P2P MMOG architectures with our reputation system in place.

Pseudospoofing
The idea of this attack is as follows. A malicious user registers with the system, behaves in a corrupt manner for a while and then re-register with the system (by quitting and rejoining the game) to re-initialize his/her reputation. With a re-initialized reputation the user can now falsely accuse another user of being malicious during a cheating detection phase.

Specifically, let $\mathcal{U}^i_{\texttt{corrupt}}$ join R^i. The initial reputation assigned to $\mathcal{U}^i_{\texttt{corrupt}}$ is Δ. After behaving maliciously, Δ will be significantly reduced. Now when $\mathcal{U}^i_{\texttt{corrupt}}$ re-registers as $\mathcal{U}^i_c$, the user's reputation will be re-initialized to Δ. With this reputation, $\mathcal{U}^i_c$ can now start to vote maliciously and accuse others of cheating. However, $\mathcal{U}^i_c$ will not gain any significant benefit from this action, because assuming the value Δ that has been chosen is sufficiently small and $|\mathcal{U}^i|$ is sufficiently large, then this malicious activity will be ineffective in our reputation system algorithm, as the sum of the other votes will out-weight $\mathcal{U}^i_c$'s vote.

Reputation Spoofing
In this type of attack, the malicious user attempts to find some vulnerabilities in the reputation algorithm and spoof the reputation values. This may be achieved by conducting reverse engineering on the software. Using our reputation system, this attack is ineffective as the reputation for a user is *not* determined nor stored by the user himself/herself, but rather is determined based on the other peers' view of this particular user. The peers in the system gain reputation when voting correctly, as their reputation will increase. Hence, by providing this mechanism, only the peers that vote correctly will benefit from this reputation system.

Whitewashing Attack
This is the common attack on the eBay online transaction system. Essentially, this means that a malicious user actively participates in the system by providing genuine items, but sometimes provides a small number of inferior goods to be sold to others [9,23]. In our scenario, consider a user $\mathcal{U}^i_k$ who is actively involved in the system by voting correctly whenever asked. Nevertheless, occasionally, this user also deliberately votes incorrectly. Note that our reputation system uses the formula $\mathsf{Rep}^i_\ell := \mathsf{Rep}^i_\ell + \xi$ to increase the value of the reputation, whilst the formula $\mathsf{Rep}^i_\ell := \mathsf{Rep}^i_\ell - c\xi$, where $c \in \{2, \cdots\}$ is used to decrease the value

of the reputation. The range of c starts from 2, which means that the 'penalty' is more severe for incorrect votes as compared to the reward given by voting correctly. For example, if $c = 10$, this refers to the case the reputation gained from 10 correct votes will completely be negated by a single incorrect vote. This way, whitewashing attacks will be ineffective.

Reputation Attacks by Collectives
This attack is achieved when malicious users know each other and they collaboratively seek to harm the system by acting as a group. An example of this kind of attacks in the P2P system is known as *shilling*. Instead of creating multiple identities as in the pseudospoofing attack, the attackers maintain several true identities to influence the voting process. This is protected by the parameter $\bar{t}$ that controls the value of the threshold in the system. Unless all users are malicious, which will make the system totally ineffective, this attack is prevented by our reputation system.

6 Conclusion

This paper addresses this vital issue by formalizing the notion of diverse P2P-based MMOG architectures into a single unifying model. We demonstrated that our formal model can be used to instantiate different P2P-based MMOG architectures. Based on this model, this paper presents a generic cheating detection mechanism that can be used to detect a number of different MMOG cheating strategies. In addition, we described a reputation system that can be used to further enhance the cheating detection process and describe its robustness against a number of attacks. Our future work includes the implementation and benchmarking of such a system in the real MMOG games.

References

1. Chang Feng, W. (ed.): Proceedings of the 3rd Workshop on Network and System Support for Games, NETGAMES 2004, Portland, Oregon, USA, 2004, August 30. ACM (2004)
2. Curtis, N., Safavi-Naini, R., Susilo, W.: X^2Rep: Enhanced Trust Semantics for the XRep Protocol. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 205–219. Springer, Heidelberg (2004)
3. Fan, L., Taylor, H., Trinder, P.: Mediator: A design framework for p2p mmogs. In: Netgames 2007 (2007)
4. Fan, L., Trinder, P., Taylor, H.: Design issues for peer-to-peer massively multiplayer online games (2009)
5. GauthierDickey, C., Lo, V.M., Zappala, D.: Using n-trees for scalable event ordering in peer-to-peer games. In: Chi Feng, W., Mayer-Patel, K. (eds.) NOSSDAV, pp. 87–92. ACM (2005)
6. GauthierDickey, C., Zappala, D., Lo, V.M.: A fully distributed architecture for massively multiplayer online games. In: Chang Feng [1], p. 171
7. Hampel, T., Bopp, T., Hinn, R.: A peer-to-peer architecture for massive multiplayer online games. In: Netgames 2006 (2006)

8. Harwood, A., Kulkani, S.: Delay sensitive identity protection in peer-to-peer online gaming environments. In: Proceedings of the 13th International Conference on Parallel and Distributed Systems, vol. 2, pp. 1–6. IEEE Computer Society, Washington, DC, USA (2007)
9. Hoffman, K., Zage, D., Nita-Rotaru, C.: A survey of attack and defense techniques for reputation systems. ACM Comput. Surv. 42, 1:1–1:31 (2009)
10. Hu, S.-Y., Chang, S.-C., Jiang, J.-R.: Voronoi state management for peer-to-peer massively multiplayer online games. In: New Interfaces for Musical Expression (2008)
11. Hu, S.Y., Liao, G.M.: Scalable peer-to-peer networked virtual environment. In: Network and System Support for Games, pp. 129–133 (2004)
12. Huang, G.-Y., Hu, S.-Y., Jiang, J.-R.: Scalable reputation management for p2p mmogs. In: Proceedings First International Workshop on Massively Multiuser Virtual Environments, pp. 4–8 (2008)
13. Iimura, T., Hazeyama, H., Kadobayashi, Y.: Zoned federation of game servers: a peer-to-peer approach to scalable multi-player online games. In: Chang Feng [1], pp. 116–120
14. Kabus, P., Buchmann, A.P.: Design of a cheat-resistant p2p online gaming system. In: Proc. of the 2nd Intl Conf. on Digital Interactive Media in Entertainment and Arts, DIMEA 2007, pp. 113–120 (2007)
15. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: WWW, pp. 640–651 (2003)
16. Knutsson, B., Lu, H., Xu, W., Hopkins, B.: Peer-to-peer support for massively multiplayer games. In: Infocom (2004)
17. Krause, S.: A case for mutual notification: A survey of p2p protocols for massively multiplayer online games. In: Netgames 2008 (2008)
18. Laurens, P., Paige, R.F., Brooke, P., Chivers, H.: A novel approach to the detection of cheating in multiplayer online games. In: 12th IEEE International Conference on Engineering Complex Computer Systems 2007, pp. 97–106 (July 2007)
19. Merabti, M., Rhalibi, A.E.: Peer-to-peer architecture and protocol for a massively multiplayer online game. In: Globecom 2004 Workshops (2004)
20. Macedonia, M.R., Zyda, M.J., Pratt, D.R., Brutzman, D.P., Barham, P.T.: Exploiting reality with multicast groups. IEEE Computer Graphics and Applications 15, 38–45 (1995)
21. Shi, X., Liu, F., Du, L., Chen, X., Xing, Y.: A cheating detection mechanism based on fuzzy reputation management of p2p mmogs
22. Vik, K.-H.: Game state and event distribution using proxy technology and application layer multicast. In: Zhang, H., Chua, T.-S., Steinmetz, R., Kankanhalli, M.S., Wilcox, L. (eds.) ACM Multimedia, pp. 1041–1042. ACM (2005)
23. Walsh, K., Sirer, E.G.: Fighting peer-to-peer spam and decoys with object reputation. In: Proceedings of the 2005 ACM SIGCOMM Workshop on Economics of Peer-to-peer Systems, P2PECON 2005, pp. 138–143 (2005)
24. Yamamoto, S., Murata, Y., Yasumoto, K., Ito, M.: A distributed event delivery method with load balancing for mmorpg. In: NETGAMES, pp. 1–8. ACM (2005)
25. Yan, J., Randell, B.: A systematic classification of cheating in online games. In: Proceedings of 4th ACM SIGCOMM Workshop on Network and System Support for Games, NetGames 2005, pp. 1–9. ACM, New York (2005)
26. Yu, L., Susilo, W., Safavi-Naini, R.: X^{2BT} Trusted Reputation System: A Robust Mechanism for P2P Networks. In: Pointcheval, D., Mu, Y., Chen, K. (eds.) CANS 2006. LNCS, vol. 4301, pp. 364–380. Springer, Heidelberg (2006)

Proof of Possession for Cloud Storage via Lagrangian Interpolation Techniques*

Łukasz Krzywiecki and Mirosław Kutyłowski

Faculty of Fundamental Problems of Technology, Wrocław University of Technology
{firstname.secondname}@pwr.wroc.pl

Abstract. We consider *Provable Data Possesion* (PDP) - a protocol which enables an owner of data stored in the cloud to verify whether this data is still available in the cloud without holding a copy of the file. We propose a PDP framework based on Lagrangian interpolation in the exponent for groups with hard discrete logarithm problem. We reuse properties of this arithmetic exploited so far in broadcast encryption systems.

Keywords: provable data possession, proof of retrievability, cloud computing, Lagrangian interpolation in the exponent, broadcast encryption.

1 Introduction

Storing Data in a Cloud. Recently, storing data in a cloud became a rapidly growing phenomenon. Despite many security problems, it becomes increasingly popular. While there are many reasons for which the users might be tempted to use clouds for storing data, providing guarantees of service quality is a hard challenge.

Clients relying on a remote data storage are interested in data durability and consistency, even if they are not (immediately) interested in the data retrieval. As a minimum condition a data owner should be given an effective way to check that

- his data is still available in the cloud in its original form, and
- once retrieved, the owner can recognize if it has been corrupted by the cloud servers.

Depending on some details a scheme fulfilling these conditions is called *PDP* (*Provable Data Possession*) or *POR* (*Proof of Retrievability*). Designing such schemes is not easy due to the following reasons:

- Typically, the data owner does not keep a copy of data stored in the cloud. So there is no reference data on the owner's side that can be used for a comparison.
- Volume of data exchanged between the user and the cloud should be minimized. In particular, downloading the data from the cloud and integrity checking can be done occasionally, but as a system routine it is practically infeasible.

A naïve solution is that: the owner generates a signature s of payload data D and stores D together with s, in order to check, the owner downloads D and s and verifiers the signature s; so whatever change is done in D the signature will be found invalid. This approach is not much useful - downloading the whole data D for every check is very costly.

* Partially supported by Foundation for Polish Science, programme "MISTRZ".

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 305–319, 2012.

Provable Data Possession. PDP is a two party client-server protocol enabling a client to check whether a file D stored in a cloud on a remote server is available there in the original form. PDP should be efficient: storage overhead and volume of data exchanged between the client and the server should be low.

A PDP scheme consists of the following procedures. *Preprocessing* run by a client creates meta-data for a given file D. Generally, the meta-data should have *small volume*, and is to be kept locally by the owner of D. The original *large* file D is transferred to the remote server for a permanent storage, possibly with some meta-data. Checking that D is stored by the server is done according to the following general scenario: In a *challenge phase* the client presents a (randomized) challenge c for a proof that a specific file D is still kept at the server. In *response* the server computes a proof P of possession data of D. Constructing the proof must require possession of the original file D, and should depend on the challenge c (to avoid replay attacks). The proof P is sent back to the client, who verifies it against the meta-data stored locally. In case of POR scheme there is a subsequent retrieval procedure during where the client may distinguish between the original file and a corrupted one retrieved from the cloud server.

Perhaps the simplest solution for PDP is based on a cryptographic hash function $\mathcal{H}$. First the client chooses at random challenges $c_1, \ldots, c_k$ and computes corresponding proofs $p_1, \ldots, p_k$, where $p_i = \mathcal{H}(c_i||D)$. During the ith challenge phase the clients sends c_i to the server. The server computes $p'_i = \mathcal{H}(c_i||D)$ and sends p'_i back to the client. If $p'_i = p_i$, then the client assumes that the server holds the original file D. Although the scheme is efficient, it has the obvious drawback that the client can check the server only k times, where k must be fixed during the preprocessing phase.

Previous Work. Since in the physical layer removing or destroying data records cannot be prevented by a client of such external systems, a proof of possession becomes very important from the point of view of a data owner.

For this reason, PDP/POR techniques were recently analyzed in many papers (see e.g. [1–12]). The notion PDP has been introduced by Ateniese et al. in [1]. According to their method, a file is divided into a number of blocks and a cryptographic tag is created for each block. The tag for the ith block m has the form $T_{i,m} = (H(W_i)g^m)^d \bmod N$,where N is and RSA number, d is a secret key of the data owner, and $H(W_i)$ is a pseudorandom value computed with a secret key of the data owner. The scheme from [1] enables to test any number of blocks in one shot without retrieving them. The trick is to compute $T = T_{i_1,m_{i_1}}^{a_1} \cdot \ldots \cdot T_{i_c,m_{i_c}}^{a_c}$ where the block numbers $i_1, \ldots, i_c$ as well as $a_1, \ldots, a_c$ are derived in a pseudorandom way from the challenge, and a sibling value $\rho = H(g_s^{a_1 m_{i_1} + \cdots + a_1 m_{i_1}})$($g_s = g^s$ is again a part of the challenge). A verifier can remove the exponent d by raising to power e, and divide the result by all factors $H(W_{i_j})$. Then the result can be raised to power s in order to get $g_s^{a_1 m_{i_1} + \cdots + a_1 m_{i_1}}$ and check it against ρ.

The biggest disadvantage of the scheme is usage of RSA numbers; this leads to relatively heavy arithmetic and somewhat long tags, challenge and response values.

A closely related concept - *proof of retrievability (POR)* - has been introduced by Juels and Kaliski in [13]. The scheme is devoted to encrypted files and the main

approach is to hide sentinels that can be used to detect file modifications. However, only a fixed number of challenges is allowed and verification has to be done by the data owner.

An important milestone for design of PDP/POR is the paper [14]. It reuses general algorithm design from [1], but the details are much different. Again, the server stores tags that mix information on file blocks with the information that can be retrieved by the data owner. The focus is on the size of the challenge and response data – they are considerably reduced compared to [1]. There are two schemes proposed. The first one is based on pseudo-random functions; its verification procedure requires the secret key of the data owner (so it is not publicly verifiable). However, the main contribution is a rigorous security proof in the standard model. The second scheme is based on BLS signatures and its security is proved under CDH assumption over bilinear groups in the random oracle model. The mechanism of the scheme depends very much on bilinear mappings.

[15] introduces a concept that has been recently substantially improved in [11]. The main idea of [11] is to create tags as commitments to polynomials in an efficient way discovered recently in [16]. Blocks of a file are treated as coefficients to polynomials, the tag for $m_{i,0}, \ldots, m_{i,s-1}$ is created via linear equations:

$$t_i := \mathtt{PRF}_{\mathtt{seed}}(\mathtt{id}, i) + \tau \sum_{j=0}^{s-1} m_{i,j} \cdot \alpha^j \bmod p \tag{1}$$

(τ is a secret value, `id` is the file identifier, `seed` is a secret seed, `PRF` is a pseudo-random function), and α is a fixed parameter, which is represented in the public key by values g^{α^j}. The proof procedure is based on polynomial commitment from [16] and uses evaluation of polynomials in the exponent instead of bilinear mapping. Due to usage of evaluation in the exponent this scheme is related to ours, however this is almost the only common technical element of both schemes.

Our Contribution. We propose a PDP/POR scheme based on techniques closely related to Broadcast Encryption. Apart from the standard goals we aim to fulfill the following conditions:

1. the solution should be based on standard algebraic operations so that it can be used on a wide range of devices,
2. design of the scheme should be as simple as possible and transparent to the users.

These conditions are related to practical implementation issues. For instance, a solution based on exotic algebraic structures is likely to require code that is never used by any other application. This increase implementation cost and reduce usability, especially for resource limited devices (like smart phones and many embedded systems). The second condition is very important for building trust. If the cryptographic mechanism behind the scheme is easy enough so that an average computer professional can understand it and estimate the strength of the proof, we no longer depend on pure trust. Last not least such solutions are better accepted by the users.

Developing a system based on another component (in our case BE) has yet another advantage. First, it might be the case that we re-use a lot of code, and therefore development costs are reduced as well as space requirements for the devices (if some routines

serve many purposes). It also reduces effort for product evaluation and audit. Composing a new solution out of known and certified products makes evaluation procedure much easier and more reliable.

We present a scheme based on Lagrangian interpolation in the exponent in a group with hard Discrete Logarithm Problem. It requires neither special properties of RSA arithmetic nor bilinear mappings as the previous techniques getting similar functionalities. Our scheme borrows many ideas from Broadcast Encryption from [17].

2 Preliminaries

Lagrangian Interpolation in the Exponent. Let $L : G \to G$ be a polynomial of degree z, and $A' = \langle (x_0, g^{rL(x_0)}), \ldots, (x_z, g^{rL(x_z)}) \rangle$ be a list of pairs where $x_i \neq x_j$ for $i \neq j$. Then we can reconstruct $g^{rL(x)}$, for an arbitrary x, by *Lagrangian interpolation in the exponent*:

$$LI_{EXP}(x, A') \stackrel{\text{def}}{=} \prod_{i=0,(x_i,.)\in A'}^{z} \left(g^{rL(x_i)}\right)^{\prod_{j=0,j\neq i}^{z} \left(\frac{x-x_j}{x_i-x_j}\right)}.$$

Note that the left right-hand side expression equals

$$g^{r\sum_{i=0,(x_i,.)\in A'}^{z} \left(L(x_i)\prod_{j=0,j\neq i}^{z} \left(\frac{x-x_j}{x_i-x_j}\right)\right)} = g^{rL(x)}.$$

The most useful property of Lagrangian interpolation in the exponent is that given z pairs $(x_1, L(x_1)), \ldots, (x_z, L(x_z))$ and additionally $(x_0, g^{rL(x_0)}), g^r$:

- it is easy to compute $g^{r\cdot L(u)}$ for an arbitrary u,
- it is infeasible to compute the values of the polynomial $\hat{g}^{L(x)}$ for $\hat{g}$ chosen at random, and in particular
- it is infeasible to reconstruct the polynomial L,

provided that group G fulfills standard hardness assumptions for discrete logarithm.

Broadcast Encryption (BE). BE is a scheme in which a single unit, called *broadcaster*, sends data to authorized users via a shared broadcast channel. The broadcast channel is accessible to everyone in its range, so BE has to ensure that:

- a non-authorized user cannot retrieve the data from the message broadcasted,
- each authorized user can retrieve this data,
- the system supports changes of the set of authorized users.

The simplest way to achieve these properties is to broadcast data in an encrypted form, where the decryption key is given to the authorized users only. The hardest design problem of BE is how to change the set of authorized users. Usually we try to encode the new key in a short broadcast message, called *enabling block*, so that only authorized users can derive this key.

Encoding based on Lagrangian interpolation is one of the fundamental techniques for BE revocation [17]. In this system a user i holds its share $(x_i, L(x_i))$ of a secret polynomial L of degree z. The new key K is transmitted in a so called *enabling block*:

$$g^{r \cdot L(0)} \cdot K, g^r, (u_1, g^{r \cdot L(u_1)}), \ldots, (u_z, g^{r \cdot L(u_z)})$$

where $u_1, \ldots, u_z$ are pairwise disjoint. So, holding one additional value $(x_i, L(x_i))$ it is easy to reconstruct $g^{r \cdot L(0)}$ via Lagrangian interpolation in the exponent, and thereby to derive K. On the other hand, if the broadcaster wants to prevent a user i to get K, he can use x_i as one of the values u_j. Then user i has only z values of the polynomial $g^{r \cdot L(x)})$ and therefore is unable to derive it. Moreover, since r is chosen for each enabling block independently at random, the values contained in one enabling block are useless for decrypting the key K from a different enabling block.

3 Outline of the Scheme

In our scheme we use a cyclic group G of a prime order q such that Discrete Logarithm Problem is hard in G. Let g be a generator of G.

A file to be stored in the cloud is divided into *blocks* and each block into z *sublocks*, where z is a scheme parameter. Each subblock is represented by a single element from G. The parameter z is chosen according to operating system requirements (preferably, a block should consist of some number of *pages* in the sense of the operating system).

Single-Block PDP: It is a PDP scheme restricted to a single block. The proofs concern *all* subblocks of a block, however the size of challenges and responses corresponds to the subblock size.

Let a block f consist of subblocks $m_1, \ldots, m_z$. The block f also corresponds to a secret polynomial L_f of degree z known to the data owner but unknown for the cloud. We may assume w.l.o.g. that the subblocks are pairwise different, e.g. each subblock contains its serial number. The tag t_i for m_i is defined as $L_f(m_i)$ and is known to the cloud. Note that the number of values of polynomial L_f given by the tags is insufficient to reconstruct the polynomial L_f – one value is missing for interpolation of L_f.

The verifier requests the cloud to compute the value $g^{rL_f(x_c)}$ when $(x_c, g^{rL_f(0)})$ and g^r are given. The number r is chosen at random, independently for each request. The element x_c is fixed and different from all possible m_i's. The requested value is obtained via Lagrangian interpolation in the exponent, it requires z group exponentiations per block on the server side. At the same time, the cloud server can derive neither $L_f(0)$ nor $L_f(x_c)$ as they are obtained in the exponent and additionally blinded by r.

If at least one m_i is lost, it is impossible to use polynomial interpolation even if $L_f(m_i)$ is available. Moreover, the scheme is strong in the information-theoretic sense: for each potential answer a there is a polynomial L' that satisfies $a = g^{rL'(x_c)}$, $L'(0) = L_f(0)$, as well as $L'(m_j) = L_f(m_j)$ for $j \neq i$, and there is an y such that $L'(y) = L_f(m_i)$. Also, it is infeasible to use values generated for challenge g^r to answer a query with the challenge $g^{r'}$ for $r' \neq r$.

Computational complexity on the server side is z group exponentiations per block. So for verification of k blocks the server has to perform kz group exponentiations.

Table 1. Sequence diagram of the PDP scheme

Client C		Server S
setup: $SK_C \longleftarrow \mathsf{Setup}(\xi)$ $T_f \longleftarrow \mathsf{TagBlock}(SK_C, m)$ erases f	$\xrightarrow{T_f}$	stores T_f
challenge: $(K_f, H) \longleftarrow \mathsf{GenChallenge}(SK_C, ID(f))$ store K_f	$\xrightarrow{H, ID(f)}$	locates T_f for $ID(f)$ $P_f \longleftarrow \mathsf{GenProof}(T_f, H)$
$\mathsf{CheckProof}(P_f, K_f)$	$\xleftarrow{P_f}$	

Table 1 provides a sequence diagram of the scheme. Note that after transferring the tagged block T_f to the cloud server S, the client C can erase f to save the local storage space. Thus C uses only short block identifier $ID(f)$ in subsequent computations. The procedures are described in Definition 1.

Definition 1. $\mathsf{Single-Block-PDP}$ *scheme is a tuple of procedures* (Setup, TagBlock, GenChallenge, GenProof, CheckProof) *where:*

- *initialization procedure* $\mathsf{Setup}(\xi)$ *is a probabilistic key generation algorithm run by the client. Its input is a security parameter* ξ*. It returns a secret keys* SK_c *of the client.*
- *procedure* $\mathsf{TagBlock}(SK_c, f)$ *is run by the client. It takes as input a secret key* SK_c *and a block* f *and returns* T_f*, a tagged version of* f*.*
- *procedure* $\mathsf{GenChallenge}(SK_c, ID(f))$ *is run by the client in order to generate a challenge for a cloud server holding* T_f*. Its input are the secret key* SK_c *and a block id* $ID(f)$*. It returns a verification value* K_f *and a challenge* H *for the server.*
- *procedure* $\mathsf{GenProof}(T_f, H)$ *is run by the cloud server in order to generate a proof of possession of* f*. Its input are the tagged blocks* T_f *of a block* f *and a challenge* H*. It returns the proof of possession* P_f *for the blocks* T_f*.*
- *procedure* $\mathsf{CheckProof}(P_f, K_f)$ *is run by the client in order to validate the proof of possession. It receives as an input the verification value* K_f *from the local memory of the client and a proof of possession* P_f *from the cloud server. It returns Accept or Reject as evaluation of* P_f*.*

A supplementary procedure $\mathsf{PubChall}(PK_f)$ *can be run by a* public *verifier aiming to check that a cloud server is holding* T_f*. The procedure receives as an input the public key* $PK_f = (K_f, H)$ *which is a chosen output from* GenChallenge *procedure.* PubChall *returns another pair* $(K_{f_{new}}, H_{new})$*, which forms a new verification value and a new challenge for the cloud server holding* T_f*.*

General PDP Scheme: The trick is to use different polynomial L_f for each block f, but with the same value at zero. Then the same challenge $g^{rL_f(0)}, g^r$ can be used for all

blocks. The client sends only one challenge altogether with block identifiers to check the possession of them by the server.

The cloud server aggregates the proofs of possession $g^{rL_f(x_c)}$ obtained for different blocks by multiplying them. The client checks that the response equals $(g^r)^{\sum_f L_f(x_c)}$.

Definition 2. $\mathsf{General-PDP-Scheme}$ *consists of procedures* $\mathsf{AggGenChallenge}$ *and* $\mathsf{AggGenProof}$*, where:*

- *the aggregation challenge generating procedure* $\mathsf{AggGenChallenge}(SK_c, F)$ *is run by the client in order to generate an aggregated challenge corresponding to many blocks for a server holding a set. It receives as an input the secret key* SK_c*, and a collection of blocks identifies* $F = \{ID(f_1), \ldots, ID(f_k)\}$*. It returns the verification value* K_f *and the short challenge* H *for the server (*$H = (x_c, g^r, (0, g^{rL(0)}))$ *in our case).*
- *the proof generating procedure* $\mathsf{AggGenProof}(F, H)$ *is run by the server in order to generate an aggregated proof of possession. It receives as an input the list of identifiers* F*, the tagged blocks* T_f *for each* $ID(f) \in F$ *and a challenge* H*. It returns the proof of possession* P_f *for all blocks of* F*.*

It is very important that the data owner does not have to store all polynomials L_f (one polynomial per block). They can be generated in a pseudorandom way from some secret seed chosen by the data owner. Therefore the local storage usage is minimized.

4 PDP via Lagrangian Interpolation

In this section we describe in detail our PDP scheme for a single block. First let us fix some preliminary issues:

- Let p, q be prime numbers, $q|p-1$, and G be a subgroup of order q in $\mathbb{Z}_p^*$, for which Discrete Logarithm Problem (DLP) is hard. Let g be a generator of G.
- C will denote a client, S will denote a cloud server.
- We shall consider storing a block f in a cloud. We assume that f is a tuple of z subblocks: $f = (m_1, \ldots, m_z)$, where each m_i contains its index i. Moreover, each m_i can be regarded as an element of $\mathbb{Z}_q$.
- For $f = (m_1, \ldots, m_z)$, T_f is a tagged version of f. It is a tuple of z pairs: $T_f = ((m_1, t_1) \ldots, (m_z, t_z))$ such that t_i is obtained via $\mathsf{TagBlock}$ described below. We assume that each t_i can be interpreted as an element of $\mathbb{Z}_q$.
- $ID(f)$ is a unique identifier for the block f. Similarly, $ID(S)$ is an identifier for the remote server S.
- $SPRNG(SK, ID(S), i, \mathbb{Z}_q)$ is a secure pseudorandom number generator that generates elements from $\mathbb{Z}_q$ in a way indistinguishable from a uniform random generator. $SK, ID(S), i$ are used as the seed.
- let $a \longleftarrow_R A$ mean that the element a is drawn from A uniformly at random.

4.1 Procedures

Procedure Setup defines system parameters for a user:

Data: system security parameter ξ
Result: group G, the master secret key SK_C of the user
begin
  choose G subgroup of $\mathbb{Z}_p^*$ of order q, such that q, p are prime, $q|p-1$, *BitLength*$(q) > \xi$, and DLP is hard in G
  $SK_C \longleftarrow_R \mathbb{Z}_q$
  return SK_C;

Procedure 1: Setup algorithm Setup(ξ)

Procedure Poly yields a secret polynomial L_f over $\mathbb{Z}_q$ for a given block f, described by the index *ID*(f). The polynomial is derived in a pseudorandom way with *SPRNG* seeded by the secret key of the client, and the block identifier *ID*(f). Thus the client can easily reconstruct L_f without the need to store polynomial L_f or f. L_f must have degree z, where z is the number of subblocks in f.

Data: the secret SK_C, the maximum number of subblocks z, the identifier *ID*(m), secure pseudorandom number generating function *SPRNG*, system security parameter ξ
Result: the polynomial $L_f(x)$
begin
  for $i = 0, \ldots, z$ **do**
    $a_i \longleftarrow$ *SPRNG*$(SK_C, ID(f), i, \mathbb{Z}_q)$
  $L_f(x) \longleftarrow \sum_{i=0}^{z} a_i x^i$
  return $L_f(x)$;

Procedure 2: Polynomial generating sub-procedure Poly

In the above procedure we silently assume that $a_z \neq 0$. If $a_z = 0$, then we run the generator until an element a_z different from 0 is obtained.

Procedure TagBlock is a tag generating procedure performed by the client. After reconstructing the secret block polynomial $L_f(x)$ via Poly sub-procedure, the client computes $t_i = L_f(m_i)$ for each subblock $m_i \in f$:

Data: a block $f = (m_1, \ldots, m_z)$, the secret SK_C generated by Setup Procedure
Result: a tagged block $T_f = ((m_1, t_1), \ldots, (m_z, t_z))$
begin
  $L_f(x) \longleftarrow$ Poly$(SK_C, z, ID(f), SPRNG)$
  foreach $m_i \in m$ **do**
    $t_i \longleftarrow L_f(m_i)$
  return $T_f = ((m_1, t_1), \ldots, (m_z, t_z))$;

Procedure 3: Tag generating procedure TagBlock(SK_c, f)

Procedure GenChallenge is executed by the client and yields a challenge for the cloud. The client reconstructs polynomial $L_f(x)$ via Poly sub-procedure. The client chooses at random an $r \in \mathbb{Z}_q$. Then, the client chooses at random x_c such that $(x_c \neq 0)$ and

$(x_c \neq m_i)$ for each subblock $m_i \in f$. The proof to be returned by the cloud equals to $g^{rL_f(x_c)}$. It is stored locally by the client as K_f. (Storing K_f is only for the sake of efficiency, as it can be recomputed after obtaining the proof from the cloud server S). Finally the user creates the challenge $H = \langle g^r, x_c, g^{rL_f(0)} \rangle$:

Data: the master secret SK_C, the block identifier *ID*(f)
Result: $(K_f, H = \left\langle g^r, x_c, g^{rL_f(0)} \right\rangle)$
begin
  $L_f(x) \longleftarrow \mathsf{Poly}(SK_C, z, ID(f), SPRNG)$
  $r \longleftarrow_R \mathbb{Z}_q$
  $x_c \longleftarrow_R \mathbb{Z}_q$ s.t. $(x_c \neq m_i)$ for each $m_i \in f$
  $K_f = g^{rL_f(x_c)}$;
  $H \longleftarrow \left\langle g^r, x_c, g^{rL_f(0)} \right\rangle$
  return (K_f, H)

Procedure 4: Generating a challenge via $\mathsf{GenChallenge}(SK_c, ID(f))$

Procedure GenProof executed by the cloud server has to generate a proof of possession of f. It takes as an input the challenge $\langle g^r, x_c, g^{rL_f(0)} \rangle$ received from the user. It constructs a set Ψ=$\{(m_i, (g^r)^{t_i}) | i = 1, \ldots, z\}$, from tagged block T_f. Then it constructs interpolation set $\Psi' = \Psi \cup \{(0, g^{rL_f(0)})\}$, interpolates the polynomial L in the exponent at x_c: $P_f = LI_{EXP}(x_c, \Psi')$.

Data: the tagged block T_f, the challenge $H = \left\langle g^r, x_c, g^{rL_f(0)} \right\rangle$
Result: the proof P_f
begin
  $\Psi = \emptyset$
  foreach $(m_i, t_i) \in T_f$ **do**
    $\Psi \longleftarrow \Psi \cup \{(m_i, (g^r)^{t_i})\}$
  $\Psi' \longleftarrow \Psi \cup \{(0, g^{rL_f(0)})\}$
  $P_f \longleftarrow LI_{EXP}(x_c, \Psi')$
  return P_f;

Procedure 5: Proof procedure $\mathsf{GenProof}(T_f, H)$

Procedure CheckProof executed by the user verifies the proof returned by the cloud with the answer K_f retained locally:

Data: P_f returned from the cloud, K_f stored locally
Result: *Accept* if the proof is correct, *Reject* otherwise
begin
  if $P_f == K_f$ **then**
    return *Accept*
  else
    return *Reject*

Procedure 6: Proof procedure $\mathsf{CheckProof}(P_f, K_f)$

4.2 Supplementary Procedures

Public Verification. In this scenario a client publishes a public key PK_f for the block f. The key PK_f, in fact, is a pair (K_f, H) returned by the GenChallenge procedure. PK_f forms an input tuple $\left\langle g^{rL_f(x_c)}, g^r, x_c, g^{rL_f(0)} \right\rangle$ for procedure PubChall. This procedure transforms these values by replacing r by $r \cdot r'$ for a random r'. Thereby, the verifier gets fresh instances indistinguishable from the result of GenChallenge. With these values the verifier can proceed in the way already described. Observe that PubChall yields the results with the same probability distribution as GenChallenge.

Data: a public key PK_f for the block f, $PK_f = \left\langle g^{rL_f(0)}, g^r, x_c, g^{rL_f(x_c)} \right\rangle$
Result: $(K_f = (g^{rL_f(x_c)})^{r'}, H = \left\langle (g^r)^{r'}, x_c, (g^{rL_f(0)})^{r'} \right\rangle)$

```
1 begin
2     r' ⟵_R Z_q
3     K_f ⟵ (g^{rL_f(x_c)})^{r'}
4     H ⟵ ⟨(g^r)^{r'}, x_c, (g^{rL_f(0)})^{r'}⟩
5     return (K_f, H)
```

Procedure 7: Public challenge generating PubChall

Aggregation of Challenges and Proofs. Here we present a modified version of algorithms that allow to aggregate challenges and proofs for many blocks (files) stored at server S, as mentioned in Sect. 3 (General PDP Scheme). Now all coefficients but a_0 in polynomials L_f are unique for each block f; while a_0 is the same for all blocks.

Data: the secret SK_C, the maximum number of subblocks z, the identifier $ID(f)$, secure pseudorandom number generating function *SPRNG*, system security parameter ξ
Result: the polynomial $L_f(x)$

```
1 begin
2     a_0 ⟵ SPRNG(SK_C, ID(S), 0, Z_q)
3     for i = 1, ..., z do
4         a_i ⟵ SPRNG(SK_C, ID(f), i, Z_q)
5     L_f(x) ⟵ Σ_{i=0}^{z} a_i x^i
6     return L_f(x);
```

Procedure 8: Polynomial generating sub-procedure AggPoly

In the modified procedure the challenge and the proof are constructed by interpolating polynomials at $x_c \neq 0$. The value g^{ra_0} is sent to the cloud server as the value $g^{rL_f(0)}$ for every single questioned block. The proof value K_f is now the product of proof values determined separately for questioned blocks.

Data: the master secret SK_C, the block identifiers $F = \{ID(f_1), \ldots, ID(f_k)\}$
Result: $(K_f, H = \left\langle g^r, x_c, g^{rL(0)} \right\rangle)$

begin
  foreach $ID(f_i) \in F$ **do**
    $L_{f_i}(x) \longleftarrow \mathsf{AggPoly}(SK_C, z, ID(f_i), SPRNG)$
  $r \longleftarrow_R \mathbb{Z}_q$
  $x_c \longleftarrow_R \mathbb{Z}_q$ s.t. $(x_c \neq m)$ for each $m \in f$
  $K_f \longleftarrow g^{rL_{f_1}(x_c)} \cdot \ldots \cdot g^{rL_{f_k}(x_c)}$
  $H \longleftarrow \langle g^r, x_c, g^{rL(0)} \rangle$
  return (K_f, H)

Procedure 9: Challenge generating AggGenChallenge

Data: The list of identifiers F, the tagged blocks T_f for each $ID(f) \in F$, the challenge $\left\langle g^r, x_c, g^{rL(0)} \right\rangle$
Result: The proof P_f

begin
  $P_f = 1$
  foreach $ID(f) \in F$ **do**
    $\Psi_f = \emptyset$
    foreach $(m_i, t_i) \in T_f$ **do**
      $\Psi_f \longleftarrow \Psi_f \cup \{(m_i, (g^r)^{t_i})\}$
    $\Psi'_f \longleftarrow \Psi_f \cup \{(0, g^{rL(0)})\}$
    $P_f \longleftarrow P_f \cdot LI_{EXP}(x_c, \Psi'_f)$
  return P;

Procedure 10: Proof procedure AggGenProof

5 Security of the Scheme

Due to lack of a space, we provide only a proof for a simplified adversary model.

Security of the proposed PDP scheme relies on assumptions: CDH and PRNG, defined formally below. The CDH assumption should allow the client to encode a challenge as $g^{rL_f(0)}$ multiple times for the same $L_f(0)$ but for different parameters r.

Definition 3 (CDH Assumption). *Let $\langle g \rangle$ be a cyclic group generated by element g of order $\operatorname{ord} g = q$. There is no efficient probabilistic algorithm $\mathcal{A}_{\mathrm{CDH}}$ that given (g, g^a, g^b) produces g^{ab}, where a, b are chosen at random from $\mathbb{Z}_q$.*

Definition 4 (PRNG Assumption). *Let SPRNG be a pseudorandom number generator that takes as an input any seed $\{0,1\}^*$ and outputs elements from $\mathbb{Z}_q$. There is no efficient probabilistic algorithm $\mathcal{A}_{PRNG}$ that distinguishes with a non-negligible advantage between two distributions $D_1 = (SPRNG(sk, 1), \ldots, SPRNG(sk, k))$ and*

$D_0 = (r_1, \ldots, r_k)$, where $sk, r_1, \ldots, r_k$ are chosen at random from $\mathbb{Z}_q$. That is, for any efficient probabilistic algorithm $\mathcal{A}_{PRNG}$ the advantage $\mathbf{Adv}(\mathcal{A}_{PRNG})$

$$\mathbf{Adv}(\mathcal{A}_{PRNG}) = \Pr[\mathcal{A}_{PRNG}(D_1) = 1] - \Pr[\mathcal{A}_{PRNG}(D_0) = 1]$$

is negligible, i.e., $\mathbf{Adv}(\mathcal{A}_{PRNG}) \leq \epsilon_{PRNG}$ for sufficiently small ϵ_{PRNG}.

Let's now analyze security of the PDP scheme from Sect. 4.1.

Definition 5. *We consider the experiment of running a forger algorithm $\mathcal{A}$ by a server S for a tagged block $T_f = ((m_1, t_1), \ldots, (m_z, t_z))$. We consider that server runs correctly through q sessions using the unmodified T_f. Then we assume w.l.o.g. that (m_1, t_1) is erased permanently (the case that only m_1 is erased can be treated similarly). To answer the subsequent challenge S invokes the algorithm $\mathcal{A}$ that takes as an input all previous knowledge S gains (i.e. results of computations and interpolations), and the truncated $T'_f = T_f \setminus \{(m_1, t_1)\}$.*

Let $I_i = \{(x_c, g^{r_i L_f(x_c)}), (0, g^{r_i L_f(0)}), (m_1, g^{r_i t_1}), \ldots, (m_z, g^{r_i t_z})\}$ denotes results of computations for the ith challenge $H_i = \langle g^{r_i}, x_c, g^{r_i L_f(0)} \rangle$).

Experiment $\mathbf{Exp}_{\mathcal{A}}$
  let $f = (m_1, \ldots, m_z)$
  let $T_f = (m_1, t_1), \ldots, (m_z, t_z)$
  For $(i = 1$ to $q)$ collect data I_i, H_i
  Erase (m_1, t_1): $T'_f = T_f \setminus \{(m_1, t_1)\}$
  Let $H = \left\langle g^r, x_c, g^{rL_f(0)} \right\rangle)$
  Run $\mathcal{A}(I_1, H_1, \ldots, I_q, H_q, T'_f, H) \rightarrow P_f$
  if $(P_f == g^{rL_f(x_c)})$ then
    return 1 else return 0

Then we define the advantage $\mathbf{Adv}(\mathcal{A})$ of the algorithm $\mathcal{A}$ in experiment $\mathbf{Exp}_{\mathcal{A}}$ as the probability $\Pr[\mathbf{Exp}_{\mathcal{A}}$ returns $1]$.

Theorem 1. *$\mathbf{Adv}(\mathcal{A})$ is negligibly small.*

Proof. According to the framework of security games [18], we construct a sequence of games against the adversary $\mathcal{A}$.

Game 0. We define Game 0 as the original attack game against $\mathcal{A}$ in the experiment $\mathbf{Exp}_{\mathcal{A}}$: Let S_0 be the event that $P_f == g^{rL_f(x_c)}$ in Game 0. Thus we have $\Pr[S_0] = \mathbf{Adv}(\mathcal{A})$.

Game 1. We define Game 1 by a slight modification of Game 0. Namely we replace pseudorandom number generator *SPRNG* in the polynomial L_f setup by a truly random draw. Thus resulting polynomial L is a random polynomial over G. The polynomial L is kept secret by the client. The other procedures utilize this polynomial. The rest of the experiment is unchanged. Let S_1 be the event that $P_f == g^{rL(x_c)}$ in Game 1.

Claim 1. $|\Pr[S_1] - \Pr[S_0]| \leq \epsilon_{PRNG}$, where ϵ_{PRNG} is the advantage of some efficient algorithm distinguishing $SPRNG$ outputs from random choices.

Indeed, for any non negligible difference between Games 0 and 1: $|\Pr[S_1] - \Pr[S_0]|$ the algorithm $\mathcal{A}$ could be used as an distinguisher $\mathcal{D}_{PRNG}$ for *SPRNG*.

Claim 2. $\Pr[S_1] \leq \epsilon_{CDH}$.

Proof of Claim 2. Assume conversely that there exists an algorithm $\mathcal{A}$ that helps to win Game 1 with non negligible probability. Then we use $\mathcal{A}$ to construct an algorithm $\mathcal{A}_{CDH}(g, g^a, g^b))$ which computes g^{ab}, thereby breaking CDH Assumption.

Algorithm $\mathcal{A}_{CDH}(g, g^a, g^b))$
- let $t_0 \longleftarrow_R \mathbb{Z}_q$, use t_0 as $L(0)$
- let $x_c \longleftarrow_R \mathbb{Z}_q$
- use g^a as $g^{L(x_c)}$
- let $f = (m_1, \ldots, m_z)$
- let $T'_f = \{(m_2, t_2), \ldots, (m_z, t_z)\}$
- For $(i = 1$ to $q)$ do:
  - $r_i \longleftarrow_R G$
  - $A \longleftarrow \{(0, g^{r_i t_0}), (x_c, (g^a)^{r_i}), (m_2, g^{r_i t_2}), \ldots, (m_z, g^{r_i t_z})\}$
  - $g^{r_i t_1} \longleftarrow LI_{EXP}(m_1, A)$
  - $I_i = \{(x_c, (g^a)^{r_i}), (0, g^{r_i t_0}), (m_1, g^{r_i t_1}), \ldots, (m_z, g^{r_i t_z})\}$
  - $H_i = \langle g^{r_i}, x_c, g^{r_i t_0} \rangle$
- Let $H = \langle g^b, x_c, (g^b)^{t_0} \rangle$
- Run $\mathcal{A}(I_1, H_1, \ldots, I_q, H_q, T'_f, H) \rightarrow P_f$
- return P_f

In the above procedure, the polynomial L is not given explicitly, but $L(0) = t_0$, $L(x_c) = a$ and $L(m_i) = t_i$ for $i = 2, \ldots, z$. Note that this definition cannot be used in the procedure in this form, as the number a is available in the exponent only (and deriving it would mean breaking Discrete Logarithm Problem). Nevertheless, in a tricky way the attacker can construct a history of the protocol that is based on values of L at the points $0, m_2, \ldots, m_z$ and x_c (note that one of these values is available in the exponent only). The problematic case for constructing the history is the value at the point m_1, but the algorithm knows the factors r_i and so the value $g^{r_i L(m_1)}$ can be obtained via Lagrangian interpolation in the exponent.

For the challenge step (when m_1 and $L(m_1)$ are already forgotten), we do not have to derive the values $g^{bL(m_i)}$ as they are not included in the challenge. It is easy to see that if $\mathcal{A}$ works as specified, then for the given inputs it returns $P_f = g^{ab}$.

Combining Claims 1 and 2, we have that $|\Pr[S_0] - \Pr[S_1]| = |\mathbf{Adv}(\mathcal{A}) - \Pr[S_1]| \leq \epsilon_{PRNG}$ and $\Pr[S_1] \leq \epsilon_{CDH}$. Therefore $\mathbf{Adv}(\mathcal{A}) \leq \epsilon_{PRNG} + \epsilon_{CDH}$. □

6 Discussion

Space Efficient Version. The tags $t_1, \ldots, t_z$ for a block need not to be stored by the cloud. An alternative procedure is that the client determines a secret s_1 such that the cloud derives $t_1, \ldots, t_z$ with a PRNG from a seed consisting of s_1, the file name and the block number. The client derives s_1 from its own secret s_0 and the cloud server name. Additionally, the client determines t_0 with a PRNG from the seed consisting of s_0 and the file name. For computing t_c - which stands for $L(x_c)$ - the client applies

Lagrangian interpolation given the pairs $(0, t_0), (m_1, t_1), \ldots, (m_z, t_z)$ and argument x_c. The value g^{t_c} encrypted symmetrically with the private key of the client is the tag of the block to be stored by the cloud server (note that retaining g^{t_c} is necessary, as it cannot be reconstructed by the client after erasing $m_1, \ldots, m_z$ from the local memory).

Performance Comparison. Below we compare our scheme (denoted by LI) with the scheme from [11] (denoted by CX).

Space Overhead - Cloud Server: LI: z group elements per block,
LI modified: 1 group element per block and 1 secret per client,
CX: 1 group element per block and z group elements per client,
space Overhead - Client: independent of the number of blocks for both LI and CX,
creating Tags: no exponentiation for both LI and CX,
challenge & Verification: LI: 3 exponentiations per block, CX: 2 exponentiations in total,
creating a Proof: LI: z+1 exponentiations per block, CX: z–1 exponentiations in total,

File Updates. Perhaps the only important disadvantage of scheme [11] is that no data update should be done directly (this problem does not occur for our scheme). Indeed, otherwise equation (1) can be used to derive α: by subtracting we can get rid of the PRF expression, by dividing such results we get rid of τ as well. However, once α become known to the cloud, then it can modify or erase files but still being able to provide correct proofs of possession.

References

1. Ateniese, G., Burns, R.C., Curtmola, R., Herring, J., Kissner, L., Peterson, Z.N.J., Song, D.X.: Provable data possession at untrusted stores. In: ACM Conference on Computer and Communications Security, pp. 598–609 (2007)
2. Curtmola, R., Khan, O., Burns, R.C., Ateniese, G.: Mr-pdp: Multiple-replica provable data possession. In: ICDCS, pp. 411–420. IEEE Computer Society (2008)
3. Chang, E.-C., Xu, J.: Remote Integrity Check with Dishonest Storage Server. In: Jajodia, S., Lopez, J. (eds.) ESORICS 2008. LNCS, vol. 5283, pp. 223–237. Springer, Heidelberg (2008)
4. Ateniese, G., Kamara, S., Katz, J.: Proofs of Storage from Homomorphic Identification Protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 319–333. Springer, Heidelberg (2009)
5. Dodis, Y., Vadhan, S.P., Wichs, D.: Proofs of Retrievability via Hardness Amplification. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 109–127. Springer, Heidelberg (2009)
6. Wang, Q., Wang, C., Li, J., Ren, K., Lou, W.: Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 355–370. Springer, Heidelberg (2009)
7. Erway, C.C., Küpçü, A., Papamanthou, C., Tamassia, R.: Dynamic provable data possession. In: ACM Conference on Computer and Communications Security, pp. 213–222 (2009)
8. Zhu, Y., Wang, H., Hu, Z., Ahn, G.J., Hu, H., Yau, S.S.: Efficient provable data possession for hybrid clouds. In: Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, pp. 756–758. ACM, New York (2010)

9. Ateniese, G., Burns, R., Curtmola, R., Herring, J., Khan, O., Kissner, L., Peterson, Z., Song, D.: Remote data checking using provable data possession. ACM Trans. Inf. Syst. Secur. 14(1), 12:1–12:34 (2011)
10. Yang, J., Wang, H., Wang, J., Tan, C., Yu, D.: Provable data possession of resource-constrained mobile devices in cloud computing. JNW 6(7), 1033–1040 (2011)
11. Xu, J., Chang, E.C.: Towards efficient provable data possession. IACR Cryptology ePrint Archive 2011, 574 (2011)
12. Shen, S.T., Tzeng, W.G.: Delegable Provable Data Possession for Remote Data in the Clouds. In: Qing, S., Susilo, W., Wang, G., Liu, D. (eds.) ICICS 2011. LNCS, vol. 7043, pp. 93–111. Springer, Heidelberg (2011)
13. Juels, A., Kaliski Jr., B.S.: Pors: proofs of retrievability for large files. In: Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS 2007, pp. 584–597. ACM, New York (2007)
14. Shacham, H., Waters, B.: Compact Proofs of Retrievability. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 90–107. Springer, Heidelberg (2008)
15. Boneh, D., Shen, E., Waters, B.: Strongly Unforgeable Signatures Based on Computational Diffie-Hellman. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 229–240. Springer, Heidelberg (2006)
16. Kate, A., Zaverucha, G.M., Goldberg, I.: Constant-Size Commitments to Polynomials and Their Applications. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 177–194. Springer, Heidelberg (2010)
17. Naor, M., Pinkas, B.: Efficient Trace and Revoke Schemes. In: Frankel, Y. (ed.) FC 2000. LNCS, vol. 1962, pp. 1–20. Springer, Heidelberg (2001)
18. Shoup, V.: Sequences of games: a tool for taming complexity in security proofs (2006), `http://www.shoup.net/papers/games.pdf`

Practical Certificateless Public Key Encryption in the Standard Model

Wenjie Yang[1], Futai Zhang[1,2], and Limin Shen[1,2]

[1] School of Computer Science and Technology,
Nanjing Normal University, 210046, China
njnuywj@163.com, {zhangfutai,shenlimin}@njnu.edu.cn
[2] Jiangsu Engineering Research Center on Information Security
and Privacy Protection Technology, Nanjing, 210097, China

Abstract. Certificateless public key cryptography was introduced to solve the key escrow problem in identity based cryptography without using public key certificates. In this paper, we investigate certificateless encryption (CLE) scheme whose security does not rely on random oracles. We present a practical CLE scheme which is proved CCA2 secure in the standard model. Our scheme can be regarded as a modification to Zhang et al.'s scheme [5] which has been confirmed insecure against type II adversaries. The newly proposed CLE scheme not only overcomes the security drawbacks of Zhang et al.'s scheme, but also preserves its most distinctive feature of a short public key length. The formal security proof shows that the new scheme is IND-CCA2 secure against Type I and Type II attackers without random oracle under the decisional truncated q-ABDHE assumption and the DBDH assumption, respectively.

Keywords: Certificateless encryption scheme, Malicious KGC attack, Standard model.

1 Introduction

In the traditional public key cryptography (PKC), a trusted party called the certification authority (CA) issues a digitally signed certificate binding the identity and the public key of a user to guarantee the authenticity of public keys. However, the issues associated with certificate management are considered costly and becomes the main difficulty for the deployment and application of traditional PKC. To simplify public key management and avoid the use of digital certificates, Shamir [1] first introduced the notion of identity-based (ID-based) cryptography (ID-PKC) in which an entity's public key is derived directly from its identity information, for example, the name, e-mail address, or IP address of the user. The private keys are fully generated by a private key generator (PKG) which inevitably introduces the key escrow problem.

Certificateless public key cryptography (CL-PKC), introduced by Al-Riyami and Paterson in 2003 [2], is intended to solve the key escrow problem which seems inherent in ID-based cryptography, while enjoys its attractive certificate free

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 320–333, 2012.

property. In CL-PKC, the key generation center (KGC) only produces partial private keys for users. A user computes his private key by combining his partial private key and a secret value chosen by himself. In this way, it gets rid of the key escrow problem in ID-PKC, and yet doesn't need the certificate to guarantee the authenticity of an entity's public key. Since the introduction of CL-PKC [2], many certificateless encryption schemes have been proposed [9–13, 15, 7, 19, 20].

As a user's full private key is decided by two secrets, a partial private key and a secret value, there are naturally two types of daversaries in CL-PKC. They are referred to as Type I and Type II adversaries respectively.

- **Key Replace Attack.** A third party (a Type I adversary) tries to impersonate a user after compromising the secret value and/or replacing the public key with some value chosen by the third party. However, it does not know the partial private key.
- **Malicious KGC Attack.** The KGC (a Type II adversary), who knows the partial private key of a user, is malicious and tries to impersonate the user. However, the KGC does not know the secret value or being able to replace the public key.

Note that, the original concept of the malicious KGC was presented by Au et al in [3]. Comparing with the traditional Type II adversaries[2], the malicious KGC can launch an attack at the very beginning of the Setup stage of the system. This means the KGC can be malicious so that it may not follow the scheme specification for generating the system parameters and the master key, while it does not actively replace a user's public key or corrupt the user's secret value. As all of the previous schemes have an implicit assumption that the KGC always generates its master public/secret key pair honestly according to scheme specification, almost all schemes proposed before [3] have been shown to be insecure against malicious KGC attacks.

To our best knowledge, the only provably secure certificateless encryption scheme against malicious KGC attack currently available is due to Libert and Quisquater [15]. Its security is proved in the random oracle model. Huang et al. suggested a generic construction of certificateless encryption schemes in [16, 17]. Their construction was claimed to be secure against malicious KGC in the standard model. However, they did not come up with a concrete secure scheme. In 2008, the first concrete certificateless encryption scheme (HL scheme) that was claimed to be secure against malicious KGC attack was proposed by Hwang et al. in [4]. Their scheme is based on Waters' identity-based encryption scheme [14]. However, in 2009, Zhang et al. [5] pointed out that HL scheme is insecure against the key replacement attack and gave a new certificateless encryption scheme (ZW scheme). As the ZW scheme is based on Gentry's identity-based encryption scheme [18], the scheme has a shorter public key. Unfortunately, Shen et al. [6] have testified the ZW scheme is insecure against an ordinary type II adversary and a malicious KGC.

In this paper, we modify the ZW scheme and put forward a probably-secure CLE scheme in the standard model. Our modified CLE scheme not only overcomes the vulnerability of the ZW scheme, but also preserves its most

distinctive feature of a short public key length. In addition, our construction is secure against Type I adversaries and a malicious KGC under the decisional truncated q-ABDHE assumption and the DBDH assumption, respectively.

The rest of this paper proceeds as follows. Some preliminary works and the security notions for CL-PKC scheme are given in *Section 2*. In *Section 3*, we propose our new CLE scheme. In *Section 4*, we give security analysis and performance analysis of our new scheme. The last section gives our concluding remarks.

2 Preliminaries

In this section, we briefly review some basic definitions used in this paper, including bilinear pairings, complexity assumptions and formal security model of certificateless public key encryption schemes.

2.1 Bilinear Pairings and Complexity Assumptions

Definition 1 (Bilinear Pairing). *Let G and G_T be the two multiplicative cyclic groups of order p for some large prime p. We say a map $\hat{e} : G \times G \to G_T$ a* bilinear *map if it satisfies the following properties:*

1. *Bilinearity: $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$ for all $u, v \in G$ and all $a, b \in Z_p^*$.*
2. *Non-degeneracy: $\hat{e}(u, u) \neq 1$ for some $u \in G$.*
3. *Computability: for any $u, v \in G$, $\hat{e}(u, v)$ can be computed efficiently.*

The following assumptions are given in $(G, G_T, q, g, \hat{e})$ described as above.

Definition 2 (Decisional Truncated q-ABDHE Assumption). *Given a vertor of $q+3$ elements $(g', g'^{\alpha^{q+2}}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q})$, there is no PPT algorithm $\mathcal{A}$ that can distinguish $\hat{e}(g, g')^{\alpha^{q+1}}$ from a random element $Z \in G_T$ with more than a negligible advantage ϵ. The advantage of $\mathcal{A}$ is defined as*

$$\begin{aligned} Adv_{\mathcal{A}}^{q-ABDHE}(k) = |Pr[A(g', g'^{\alpha^{q+2}}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}, \hat{e}(g, g')^{\alpha^{q+1}})] = 0 \\ -Pr[A(g', g'^{\alpha^{q+2}}, g, g^{\alpha}, g^{\alpha^2}, ..., g^{\alpha^q}, Z] = 0| \end{aligned}$$

where the probability is taken over the random choice of generator $(g, g' \in G)$, the random choice of $\alpha \in Z_p$, the random choice of $Z \in G_T$, and the random bits consumed by $\mathcal{A}$.

Definition 3 (Decisional Bilinear Diffie-Hellman (DBDH) Assumption). *Let $a, b, c, z \in Z_p^*$ be chosen at random and g be a generator of G. The (t, ϵ)-DBDH assumption is that no probabilistic polynomial-time algorithm $\mathcal{A}$ can distinguish the tuple $(A = g^a, B = g^b, C = g^c, \hat{e}(g, g)^{abc})$ from the tuple $(A = g^a, B = g^b, C = g^c, \hat{e}(g, g)^z)$ with a non-negligible advantage ϵ in time t, where the advantage of $\mathcal{A}$ is defined as*

$$\begin{aligned} Adv_{\mathcal{A}}^{DBDH}(k) = |Pr[\mathcal{A}(g, g^a, g^b, g^c, \hat{e}(g, g)^{abc}) = 1] \\ -Pr[\mathcal{A}(g, g^a, g^b, g^c, \hat{e}(g, g)^z) = 1]| \end{aligned}$$

2.2 Security Definitions and Notations

A certificateless public-key encryption scheme is defined by seven probabilistic, polynomial-time algorithms:

- ***Setup***(k)***:*** takes as input a security parameter k and returns the system master public key mpk (which includes the public parameters)and the system master secret key msk. This algorithm is run by a KGC to initially set up a certificateless system.
- ***ExtPPK***(mpk, msk, ID)***:*** takes as input the master public key mpk, the master private key msk, and an identiy $ID \in \{0,1\}^*$. It outputs a partial private key d_{ID} for the user with identity ID. This algorithm is run by a KGC once for each user, and the corresponding partial private key is distributed to that user in a suitably secure manner.
- ***SetSecV***(mpk, ID)***:*** given the master public key mpk and a user's identity ID as input, it outputs a secret value sv_{ID} for that user. This algorithm is run once by each user.
- ***SetPubK***(mpk, sv_{ID})***:*** given the master public key mpk and the secret value sv_{ID} of a user with identity ID, this algorithm outputs a public key PK_{ID} for that user. This algorithm is run once by the user and the resulting public key is widely and freely distributed.
- ***SetPriK***(mpk, sv_{ID}, d_{ID})***:*** takes as input the master public key mpk, a user's partial private key d_{ID} and its secret value sv_{ID}. It outputs the full private key SK_{ID} for that user. This algorithm is run once by the user.
- ***Encrypt***(mpk, PK_{ID}, ID, M)***:*** given a plaintext M, the master public key mpk, a receiver's public key PK_{ID} and identity ID as input, a sender runs this PPT algorithm to create a ciphertext C.
- ***Decrypt***(mpk, SK_{ID}, C)***:*** given the master public key mpk, a user's private key SK_{ID}, and a ciphertext C as input, the user as a recipient runs this deterministic algorithm to get a decryption σ, which is either a plaintext message or a "reject" message.

IND-CLE-CCA2 Security:

As defined in [3], we allow that the type II adversary can maliciously generate a pair of master public/private key by setting some trapdoors, which matches better with the original spirit [2] of the CL-PKC.

Definition 4 (IND-CLE-CCA2 Secure). *A certificateless public key encryption scheme is semantically secure against an adaptive chosen message attack (IND-CLE-CCA2 secure) if no polynomially bounded adversary $\mathcal{A}$ of Type I or Type II has a non-negligible advantage in the following game played against the challenger:*

(1). If $\mathcal{A}$ is a Type I attacker, the challenger takes a security parameter k and runs the Setup algorithm. It gives $\mathcal{A}$ the master public key mpk, and the challenger keeps the master secret key msk to itself. If $\mathcal{A}$ is a Type II attacker, the challenger invokes the adversary $\mathcal{A}$ to get mpk and msk.

(2). $\mathcal{A}$ issues a sequence of requests described as follows:

Public Key Request $\mathcal{O}_{PubO}(ID)$: On receiving such a query, the challenger responds by running algorithm SetPubK to generate the public key PK_{ID} (first running SetSecV if necessary).

Partial Private Key Extract Request $\mathcal{O}_{ParO}(ID)$: On receiving such a query, the challenger responds by running algorithm ExtPPK to generate the partial private key d_{ID}. Note that this is only useful for a Type I attacker.

Private Key Request $\mathcal{O}_{PriO}(ID)$: On receiving such a query, if the public key has not been replaced, then the challenger can respond by running algorithm SetPriK to generate the private key SK_{ID}. Note that if the corresponding public key has been replaced, when making the private key request query, the attacker $\mathcal{A}$ is required to provide his secret value sv_{ID} to the challenger.

Public Key Replace Request $\mathcal{O}_{RepO}(ID, PK_{ID})$: Such a query allows the adversary $\mathcal{A}$ to replace the public key of a user ID with any value PK_{ID} of its choice. The new value will be recorded and will be used by the challenger in the coming computations or responses to the adversary's queries. Note that only a Type I attacker is allowed to issue such a query.

Decryption Request $\mathcal{O}_{Dec}(C, PK_{ID}, ID)$: On receiving such a query, the challenger returns the correct decryption of the ciphertext C under identity ID and public key PK_{ID}. Note that if the corresponding public key has been replaced, the Type I attacker is required to provide the corresponding secret value sv_{ID} to the challenger $\mathcal{C}$ in order to correctly decrypt the ciphertext under the replaced public key.

(3). Once $\mathcal{A}$ decides that Phase 1 is over, it outputs a challenge identity ID^* with public key PK_{ID^*} and two plaintexts (m_0, m_1). Note that $\mathcal{A}$ is not allowed to know the private key of ID^* in any way. The challenger now picks a random bit b from $\{0, 1\}$ and computes the challenge ciphertext C^* of m_b under the current public key PK_{ID^*} for the corresponding user ID^*. Then, C^* is delivered to $\mathcal{A}$.

(4). $\mathcal{A}$ makes a new sequence of queries as in Step(2). In addition, there are several natural restrictions on $\mathcal{A}$:

✓ $\mathcal{A}$ can not extract the private key for the identity ID^* at any time,

✓ $\mathcal{A}$ can not make a decryption query on $\mathcal{O}_{Dec}(C^*, PK_{ID^*}, ID^*)$,

✓ $\mathcal{A}_I$ can not request the private key for any identity whose public has already been replaced,

✓ $\mathcal{A}_I$ can not extract the partial private key for ID^* if it has replaced the public key for ID^* before the challenge phase.

(5). $\mathcal{A}$ wins the game if and only if $b' = b$. We define $\mathcal{A}$'s advantage in this game to be

$$Adv_{CLE}^{IND-CLE-CCA2}(\mathcal{A}) = |Pr[b' = b] - \frac{1}{2}|$$

3 Our CLE Scheme

Let G and G_T be groups of prime order p, and let $\hat{e} : G \times G \to G_T$ be a bilinear map.

- ***Setup***(k)***:*** The KGC picks random generators $g, h_1, h_2, h_3 \in G$ and a random $\alpha \in Z_p^*$ and computes $g_1 = g^\alpha \in G$. The KGC then chooses a hash function H from a family of universal one-way hash functions. The master public key is $mpk = (p, G, G_T, \hat{e}, H, g, g_1, h_1, h_2, h_3)$ and the master secret key is $msk = \alpha$.
- ***ExtPPK***(mpk, msk, ID)***:*** The KGC randomly picks $r_1, r_2, r_3 \in Z_p$ and computes

$$h_{ID,i} = (h_i g^{-r_i})^{\frac{1}{\alpha - ID}}, i = 1, 2, 3$$

  Finally, the KGC returns $d_{ID} = (r_1, h_{ID,1}, r_2, h_{ID,2}, r_3, h_{ID,3})$ as the partial private key.
- ***SetSecV***(mpk, ID)***:*** The user picks $x_0, x_1 \in Z_p$ at random and returns $sv_{ID} = (sk_0, sk_1) = (x_0, x_1)$. as the secret value
- ***SetPubK***(mpk, sv_{ID})***:*** The user computes and publishes $PK_{ID} = (pk_0, pk_1) = (g^{x_0}, g^{x_1})$ as the public key.
- ***SetPriK***(mpk, sv_{ID}, d_{ID})***:*** The user sets

$$SK_{ID} = (sv_{ID}, d_{ID}) = (x_0, x_1, r_1, h_{ID,1}, r_2, h_{ID,2}, r_3, h_{ID,3})$$

  as the corresponding private key.
- ***Encrypt***(mpk, PK_{ID}, ID, m)***:*** To encrypt the message $m \in G_T$, parse PK_{ID} as $(pk_0, pk_1) = (g^{x_0}, g^{x_1})$, then choose $s, t \in_R Z_p^*$ and compute the ciphertext as following.

$$\begin{aligned} C = & (C_0, C_1, C_2, C_3, C_4) \\ = & (g_1^s g^{-IDs}, \hat{e}(g,g)^s, \hat{e}(g,g)^t, m \cdot \hat{e}(pk_0, pk_1)^{-t} \cdot \hat{e}(g, h_1)^{-s}, \hat{e}(g, h_2)^s \hat{e}(g, h_3)^{s\beta}) \end{aligned}$$

  where $\beta = H(C_0, C_1, C_2, C_3, ID, PK_{ID}, \hat{e}(pk_0, g)^t)$.
- ***Decrypt***(mpk, SK_{ID}, C)***:*** Parse C as $(C_0, C_1, C_2, C_3, C_4)$ and the private key SK_{ID} as $(x_0, x_1, r_1, h_{ID,1}, r_2, h_{ID,2}, r_3, h_{ID,3})$. Compute in sequence $K = C_2^{x_0} = \hat{e}(pk_0, g)^t$ and $\beta = H(C_0, C_1, C_2, C_3, ID, PK_{ID}, K)$, then check whether

$$C_4 \stackrel{?}{=} \hat{e}(C_0, h_{ID,2} h_{ID,3}^\beta) C_1^{r_2 + r_3 \beta}$$

  Reject C if the equality does not hold. Otherwise, return

$$m = C_2 \cdot K^{x_1} \cdot \hat{e}(C_0, h_{ID,1}) C_1^{r_1}$$

4 Analysis of the Scheme

4.1 Correctness

The correctness of the scheme can be directly verified by the following equations.

$$\begin{aligned} & \hat{e}(C_0, h_{ID,2} h_{ID,3}^\beta) C_1^{r_2 + r_3 \beta} \\ = & \hat{e}(g_1^s g^{-IDs}, (h_2 g^{-r_2})^{\frac{1}{\alpha - ID}} \cdot (h_3 g^{-r_3})^{\frac{\beta}{\alpha - ID}}) \hat{e}(g,g)^{s(r_2 + r_3\beta)} \\ = & \hat{e}(g^s, h_2 h_3^\beta) \hat{e}(g^s, g^{-r_2 - r_3\beta}) \hat{e}(g,g)^{s(r_2 + r_3\beta)} \\ = & \hat{e}(g, h_2)^s \hat{e}(g, h_3)^{s\beta} \\ = & C_4, \end{aligned}$$

and

$$\begin{aligned} & C_2 \cdot K^{x_1} \cdot \hat{e}(C_0, h_{ID,1}) C_1^{r_1} \\ &= C_2 \cdot \hat{e}(pk_0, g)^{tx_1} \cdot \hat{e}(g_1^s g^{-IDs}, (h_1 g^{-r_1})^{\frac{1}{\alpha - ID}}) \hat{e}(g, g)^{sr_1} \\ &= C_2 \cdot \hat{e}(pk_0, pk_1)^t \cdot \hat{e}(g^s, h_1) \\ &= m \cdot \hat{e}(pk_0, pk_1)^{-t} \cdot \hat{e}(g, h_1)^{-s} \cdot \hat{e}(pk_0, pk_1)^t \cdot \hat{e}(g, h_1)^s \\ &= m. \end{aligned}$$

4.2 Security Analysis

In this section we prove in the standard model that our construction above is secure against Type I adversaries and malicious KGCs under the decisional truncated q-ABDHE assumption and the DBDH assumption respectively.

Note that, in our security proofs, to avoid collision and consistently respond to the adversary's queries, the simulator $\mathcal{C}$ maintains the database $\mathcal{DB}$ which is initially empty. The database $\mathcal{DB}$ consists of the following four tables:

(1) PublicKeyList: (ID_i, PK_{ID_i})
(2) PartialPrivateKeyList: (ID_i, d_{ID_i})
(3) UserSecretValueList: $(ID_i, PK_{ID_i}, sv_{ID_i})$
(4) PrivateKeyList: $(ID_i, PK_{ID_i}, SK_{ID_i})$, where $SK_{ID_i} = (sv_{ID_i}, d_{ID_i})$.

Theorem 1. *Let $q = q_{Par} + 2$. Suppose $\mathcal{A}_I$ makes at most q_{Pub} times public key request queries, q_{Par} times partial private key extraction queries, q_{Pri} times private key request queries, q_{Rep} times public key replacement queries, q_{Dec} times decryption queries, and assume the decisional truncated (t', ϵ', q)-ABDHE assumption holds for $(G, G_T, \hat{e})$. Then, our proposed CLE scheme is $(t, \epsilon, q_{Pub}, q_{Par}, q_{Pri}, q_{Rep}, q_{Dec})$-IND-CCA2 secure against the Type I adversaries, where $\epsilon' \geq \epsilon - q_{Dec}/p$, $t' = t + \mathcal{O}((q \cdot q_{Par} + q_{Pub} + q_{Dec})t_e + q_{Dec}t_m + q_{Dec}t_p)$, where t_e, t_m and t_p are the time for an exponentiation, a multiplication and a pairing computation in G_T respectively.*

Proof. Assume that there exists a Type I attacker $\mathcal{A}_\mathcal{I}$ who breaks the IND-CCA2 security of our proposed scheme. We construct an algorithm, $\mathcal{C}$, which solves the decisional truncated q-ABDHE problem making use of $\mathcal{A}_\mathcal{I}$'s ability in breaking the encryption scheme. The algorithm $\mathcal{C}$ takes a random decisional truncated q-ABDHE challenge $(G, G_T, g', g'_{q+2}, g, g_1, g_2, ..., g_q, Z)$ as input, where Z is either $\hat{e}(g_{q+1}, g')$ or a random element of G_T (recall that $g_i = g^{\alpha^i}$). Algorithm $\mathcal{C}$ simulates the challenger of $\mathcal{A}_\mathcal{I}$ as follows:

Setup: $\mathcal{C}$ first generates random polynomials $f_i(x) \in Z_p[x]$ of degree q for $i \in \{1, 2, 3\}$ and sets $h_i = g^{f_i(\alpha)}$, deriving h_i from $(g, g_1, ..., g_q)$. It chooses H from a family of universal one-way hash functions. Then $\mathcal{C}$ sends the master public parameters $(p, G, G_T, \hat{e}, H, g, g_1, h_1, h_2, h_3)$ to the adversary $\mathcal{A}_\mathcal{I}$. Since α, g, $f_i(x)(i = 1, 2, 3)$ are chosen uniformly at random, h_1, h_2 and h_3 are uniformly random and the master public key has a distribution identical to that in the actual construction.

Phase 1: $\mathcal{A}_{\mathcal{I}}$ makes some queries, $\mathcal{C}$ plays the role of $\mathcal{A}_{\mathcal{I}}$'s challenger and answers the queries as follows:

- **Public Key Request** $\mathcal{O}_{PubO}(ID)$**:** With the user's ID and the master public key mpk as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1). If (ID, PK_{ID}) exists in PublicKeyList, return PK_{ID} as the answer.
  (2). Otherwise, first pick $x_0, x_1 \in_R Z_p^*$, set $sv_{ID} = (x_0, x_1)$ and compute $PK_{ID} = (pk_0, pk_1) = (g^{x_0}, g^{x_1})$, then add (ID, PK_{ID}, sv_{ID}) to UserSecretValueList and (ID, PK_{ID}) to PublicKeyList repectively. Finally return PK_{ID} as the answer.
- **Partial Private Key Request** $\mathcal{O}_{ParO}(ID)$**:** With the user's ID, the master public key mpk, and the master secret key msk as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1). Search PartialParvateKeyList for a tuple $(ID, r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$. If it exists, return $(r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$ as the answer.
  (2). Otherwise, compute the partial private key as follows:
  - ✓ If $g_1 = g^{ID}$ (i.e. $\alpha = ID$), then $\mathcal{C}$ uses α to solve the decisional truncated q-ABDHE problem immediately.
  - ✓ Otherwise, let $F_{ID,i}(x)$ denote the $q-1$ degree polynomials $(f_i(x) - f_i(ID))/(x - ID), i = 1, 2, 3$. Then, $\mathcal{C}$ sets

$$\begin{aligned}(r_{ID,i}, h_{ID,i}) &= (f_i(ID), g^{F_{ID,i}(\alpha)}) \\ &= (f_i(ID), g^{(f_i(\alpha)-f_i(ID))/(\alpha-ID)}) \\ &= (f_i(ID), (g^{(f_i(\alpha)-f_i(ID))})^{1/(\alpha-ID)}) \\ &= (f_i(ID), (h_1 g^{-f_i(ID)})^{1/(\alpha-ID)}),\end{aligned}$$

  and $d_{ID} = (r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$. Finally, add (ID, d_{ID}) to PartialPrivateKeyList and return $(r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$ as the answer.
- **Private Key Request** $\mathcal{O}_{PriO}(ID)$**:** With the user's ID, the user's secret value sv_{ID}, the user's partial private key d_{ID}, and the master public key mpk as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1). Search PrivateKeyList for a tuple $(ID, PK_{ID}, x_0, x_1, r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$. If it exists, $\mathcal{C}$ returns $(x_0, x_1, r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$ as the answer.
  (2). Otherwise, first run the simulation algorithm for public key request taking ID as the input to get two tuples (ID, PK_{ID}) in PublicKeyList, and (ID, PK_{ID}, sv_{ID}) in UserSecretValueList. Then, run the above simulation algorithm for partial private key request to get the corresponding partial private key $d_{ID} = (r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$. Finally, add the tuple $(ID, PK_{ID}, sv_{ID}, d_{ID})$ into PrivateKeyList and return (sv_{ID}, d_{ID}) as the answer. Note that if the user's public key has been replaced, the attacker is required to provide the corresponding secret value sv_{ID} to the simulator.
- **Public Key Replace Request** $\mathcal{O}_{RepO}(ID, PK_{ID})$**:** With an identity ID and a public key PK_{ID} as input, replaces the associated public key in PublicKeyList with PK_{ID}.

- **Decryption Request** $\mathcal{O}_{DecO}(C, PK_{ID}, ID)$**:** With a ciphertext C, the master public key mpk, the user's public key PK_{ID}, and an identity ID as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1). Search PrivateKeyList for a tuple $(ID, PK_{ID}, x_0, x_1, r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$. If such a tuple exists, check the validity of the public key and the ciphertext. If either is invalid, then return "invalid ciphertext" and abort; otherwise, decrypt the ciphertext as normal using the corresponding private key.
  (2). If the tuple does not exist, run the partial private key extraction query as above to get $(ID, r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$, then decrypt the ciphertext as normal using the private key $(x_0, x_1, r_{ID,i}, h_{ID,i}, i = 1, 2, 3)$. Note that if the user's public key has been replaced, the adversary $\mathcal{A}_{\mathcal{I}}$ is required to provide the corresponding secret value sv_{ID} in order to make the simulator $\mathcal{C}$ correctly complete the decryption request.

Challenge: On receiving a challenge query $(ID^*, PK_{ID^*}, M_0, M_1)$, $\mathcal{C}$ performs as follows:

(1) If $g_1 = g^{ID^*}$ (i.e. $\alpha = ID^*$), then $\mathcal{C}$ uses α to solve the decisional truncated q-ABDHE problem immediately.
(2) Otherwise, search the PartialPrivateKeyList taking (ID^*, PK_{ID^*}) as input.

✓ If there is a tuple $(ID^*, r_{ID^*,i}, h_{ID^*,i}, i = 1, 2, 3)$ in PartialPrivateKeyList, then compute the challenge ciphertext under the corresponding public key PK_{ID^*} as follows:
Select at random $b \in \{0, 1\}$ and $t \in Z_p^*$. Let $f(x) = x^{q+2}$, $F(x) = (f(x) - f(ID^*))/(x - ID^*)$ which is a polynomial of degree $q + 1$, and F_i be the coefficient of x^i in $F(x)$. Set

$$C_0^* = g'^{f(\alpha)-f(ID^*)}, C_1^* = Z \cdot \hat{e}(g', \prod_{i=0}^{q} g^{F_i \alpha^i}), C_2^* = \hat{e}(g, g)^t,$$

$$C_3^* = M_b \cdot \hat{e}(pk_0, pk_1)^{-t} \cdot \hat{e}(C_0, h_1^{-1}) \cdot C_1^{-r_1},$$

$$\beta = H(C_0^*, C_1^*, C_2^*, ID^*, PK_{ID^*}, \hat{e}(pk_0, g)^t),$$

$$C_4^* = \hat{e}(C_0^*, h_{ID^*,2} h_{ID^*,3}^{\beta}) C_1^{*(r_{ID^*,2} + r_{ID^*,3}\beta)}$$

Finally, $\mathcal{C}$ returns $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$ as the challenge ciphertext.

✓ Otherwise, run the partial private key extraction query taking (ID^*, PK_{ID^*}) as input to get the corresponding partial private key $(ID^*, r_{ID^*,i} h_{ID^*,i}, i = 1, 2, 3)$ and compute the challenge ciphertext under the corresponding public key PK_{ID^*} as the above case.

Phase 2: In this phase, $\mathcal{A}_{\mathcal{I}}$ makes a new sequence of queries, and $\mathcal{C}$ responds as in Phase 1.

Guess: Finally, the adversary $\mathcal{A}_{\mathcal{I}}$ outputs a guess b'. If $b' = b$, $\mathcal{C}$ answers 1 indicating that $Z = \hat{e}(g', g_{q+1})$; otherwise, $\mathcal{C}$ answers 0 to indicate Z as a random element of G_T.
In addition, there are several natural restrictions on $\mathcal{A}_{\mathcal{I}}$:

(1) At any time, ID^* has not been submitted to the Private key request.
(2) In Phase 2, C^* has not been submitted to the Decryption request for the combination (ID^*, PK_{ID^*}) under which M_b was encrypted.
(3) With a Type I attacker, ID^* also has not been submitted to both the Public key replace request before the challenge phase and the Partial private key extract request at any phase.

Analysis: This completes the description of simulation. It remains only to analyze the success probability and running time of $\mathcal{C}$. Analogy to [18], we can obtain that the success probability of $\mathcal{C}$ is $\epsilon' \geq \epsilon - q_{Dec}/p$ and the total running time is $t' = t + \mathcal{O}((q \cdot q_{Par} + q_{Pub} + q_{Dec})t_e + q_{Dec}t_m + q_{Dec}t_p)$. Thus, the theorem follows.

Theorem 2. *Suppose a malicious KGC ($\mathcal{A}_{\mathcal{II}}$) makes at most q_{Pub} times public key request queries, q_{Pri} times private key extraction queries, q_{Dec} times decryption queries and assume the Decisional bilinear Diffie-Hellman (t', ϵ')-DBDH assumption holds for $(G, G_T, \hat{e})$. Then, our modified CLE scheme is $(t, \epsilon, q_{Pub}, q_{Pri}, q_{Dec})$-IND-CCA2 secure against malicious KGCs, where $t' = t + \mathcal{O}((q_{Pri} + q_{Dec})t_m + (q_{Pub} + q_{Pri} + q_{Dec})t_e + q_{Dec}q_p)$, $\epsilon' \geq \frac{\epsilon}{q_{Pub}}$ where t_m, t_e and t_p are the time for a multiplication, an exponentiation and a pairing computation in G_T respectively.*

Proof. Here, we show that our construction above is secure against a malicious KGC under the DBDH assumption in the standard model. In other words, if there exists a malicious KGC who breaks the IND-CCA2 security of our modified scheme described above. We then can obtain an algorithm, $\mathcal{C}$, which solves the DBDH problem making use of $\mathcal{A}_{\mathcal{II}}$'s ability in breaking our scheme as follows.

Setup:The algorithm $\mathcal{C}$ firstly invokes $\mathcal{A}_{II}$ to generate the master secret key $msk = \alpha$ and the public parameters $params = (G, G_T, \hat{e}, H, g, g_1, h_1, h_2, h_3)$. Note that, in order to launch the Type II attack more easily, the adversary $\mathcal{A}_{II}$ is allowed to embed extra trapdoors (e.g. $h_i = g^{v_i}$, where $v_i \in_R Z_p^*$) in the system parameters according to their wishes in our security proofs. However, these extra trapdoors are kept secret to the algorithm $\mathcal{C}$. In addition, this specifically generated system parameters are computationally indistinguishable from an honestly generated system parameters.

Meanwhile, the algorithm $\mathcal{C}$ takes a random DBDH challenge (g, g^a, g^b, g^c, Z) as input, where Z is either $\hat{e}(g, g)^{abc}$ or a random element of G_T. Then, $\mathcal{C}$ chooses an index I uniformly at random with $1 \leq I \leq q_{pub}$.

Phase 1: $\mathcal{A}_{\mathcal{II}}$ makes some queries to the Public key request, Private key request and Decryption request. $\mathcal{C}$ plays the role of the $\mathcal{A}_{\mathcal{II}}$'s challenger and responds the queries as follows:

- **Public Key Request** $\mathcal{O}_{PubO}(ID)$**:** With the user's ID and the master public key mpk as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1) If (ID_i, PK_{ID_i}) exists in the PublicKeyList, return PK_{ID_i} as the answer.
  (2) Otherwise if $i \neq I$, $\mathcal{C}$ picks a random $x_{i0}, x_{i1} \in Z_p^*$, sets $sv_{ID_i} = (x_{i0}, x_{i1})$, and adds (ID_i, sv_{ID_i}) to UserSecretKeyList. Then $\mathcal{C}$ computes $PK_{ID_i} = (pk_{i0}, pk_{i1}) = (g^{x_{i0}}, g^{x_{i1}})$ and adds (ID_i, PK_{ID_i}) to PublicKeyList. Finally, $\mathcal{C}$ returns PK_{ID_i} as the answer.
  (3) If $i = I$, set $PK_{ID_I} = (pk_{I0}, pk_{I1}) = (g^a, g^b)$ and add (ID_I, PK_{ID_I}) to the PublicKeyList. Return $PK_{ID_I} = (pk_{I0}, pk_{I1})$ as the answer.
- **Private Key Request** $\mathcal{O}_{PriO}(ID_i, mpk)$**:** With the user's ID_i and the master public key mpk as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1) If there is a tuple of the form $(ID_i, PK_{ID_i}, SK_{ID_i})$ on the PrivateKeyList, $\mathcal{C}$ returns SK_{ID_i} as the answer.
  (2) Otherwise, if $i \neq I$, the algorithm $\mathcal{C}$ first runs the public key request query taking ID_i as input to get a tuple (ID_i, sv_{ID_i}) in the UserSecretValueList. Then, $\mathcal{C}$ invokes the algorithm $ExtPPK(mpk, msk, ID_i)$ to get the partial private key d_{ID_i} and adds the tuple (ID_i, d_{ID_i}) to the PartialPrivateKeyList. $\mathcal{C}$ finally adds $(ID_i, PK_{ID_i}, SK_{ID_i})$ to the PrivateKeyList and returns SK_{ID_i} as the answer.
  (3) If $i = I$, $\mathcal{C}$ aborts.
- **Decryption Request** $\mathcal{O}_{DecO}(C, PK_{ID}, ID)$**:** With a ciphertext C, the master public key mpk, the user's public key PK_{ID}, and an identity ID as input, $\mathcal{C}$ simulates the corresponding challenger as follows:
  (1) If $i \neq I$, the algorithm $\mathcal{C}$ searches the PrivateKeyList or runs the private key extraction query for a tuple $(ID_i, PK_{ID_i}, SK_{ID_i})$, then checks the validity of the ciphertext. If it is valid, $\mathcal{C}$ decrypts the ciphertext C by using SK_{ID_i}. Otherwise, return "invalid ciphertext".
  (2) If $i = I$, $\mathcal{C}$ aborts.

Challenge: On receiving a challenge query $(ID^*, PK_{ID^*}, M_0, M_1)$, if $ID^* \neq ID_I$, $\mathcal{C}$ aborts. Otherwise, $\mathcal{C}$ randomly picks $b \in \{0, 1\}$, $s \in Z_p^*$ and computes the challenge ciphertext as follows:

$$C_0^* = g_1^s g^{-ID^* s}, \quad C_1^* = \hat{e}(g, g)^s, \quad C_2^* = \hat{e}(g^c, g), \quad C_3^* = M_b \cdot Z^{-1} \cdot \hat{e}(g, h_1)^{-s}$$

$$\beta = H(C_0^*, C_1^*, C_2^*, ID^*, PK_{ID^*}, \hat{e}(g^a, g^c)), \qquad C_4^* = \hat{e}(g, h_2)^s \hat{e}(g, h_3)^{s\beta}$$

Finally, the algorithm $\mathcal{C}$ returns the ciphertext $C^* = (C_0^*, C_1^*, C_2^*, C_3^*, C_4^*)$ to $\mathcal{A}_{\mathcal{II}}$ as the challenge ciphertext.

Phase 2: In this phase, $\mathcal{A}_{\mathcal{II}}$ makes a new sequence of queries, and $\mathcal{C}$ responds as in Phase 1.

Guess: Finally, the adversary $\mathcal{A}_{\mathcal{II}}$ outputs a guess $b' \in \{0, 1\}$. If $b' = b$, $\mathcal{C}$ answers 1 indicating $Z = \hat{e}(g, g)^{abc}$, otherwise guesses Z as a random element of G_T.

In addition, there are several natural restrictions on $\mathcal{A}_{\mathcal{II}}$:

(1) At any time, ID^* has not been submitted to the Private key request.
(2) In Phase 2, C^* has not been submitted to the Decryption request for the combination (ID^*, PK_{ID^*}) under which M_b was encrypted.

Probability Analysis: If the simulation does not abort, $ID^* = ID_I$. This happens with probability $\frac{1}{q_{Pub}}$ since $\mathcal{A}_{II}$ can make q_{Pub} different public key request queries. If the simulation does not abort, the success probability of $\mathcal{A}_{II}$ is analyzed as follows. Suppose (g, g^a, g^b, g^c, Z) is not a valid DBDH instance, then the challenge ciphertext contains no valid information about the message and thus the probability of the adversary winning will be 1/2. Suppose (g, g^a, g^b, g^c, Z) is a valid DBDH instance, then by our assumption, the adversary makes the correct guess with probability $\frac{1}{2} + \epsilon$. Thus, the advantage of $\mathcal{C}$ is at least $\frac{\epsilon}{q_{Pub}}$.

Time Complexity Analysis: In the simulation, $\mathcal{C}$'s overhead is dominated by the exponentiation, multiplication and pairing operations. Thus, we will analyze the time complexity as follows.

There are $\mathcal{O}(1)$ multiplications and $\mathcal{O}(1)$ exponentiations in the private key request stage. The computational costs for the decryption query are $\mathcal{O}(1)$ multiplications, $\mathcal{O}(1)$ exponentiations and $\mathcal{O}(1)$ pairings. There are $\mathcal{O}(1)$ exponentiations in the public key query. The time complexity of $\mathcal{C}$ is $t+\mathcal{O}((q_{Pri}+q_{Dec})t_m + (q_{Pub} + q_{Pri} + q_{Dec})t_e + q_{Dec}q_p)$.

4.3 Performance Analysis

In this subsection, we compare the major overhead and security properties of our scheme with those of existing schemes that are claimed to be secure against malicious KGC attak in Table 1. For ease of analysis, we only consider the costly operations which include multiplications in $G_T(G)$, exponentiations in $G_T(G)$, and pairing operations in the Encrypt and Decrypt phases. For the security properties, we consider the confidentiality against the Type I adversaries and the Type II adversaries respectively. In addition, the form $m(+n)$ denotes the total number of pairing operations is $m + n$ where n is the number of pairings that are independent of the message and can be pre-computed.

Table 1. Performance analysis

	Multiplications	Exponentiations	Pairing	Type I	Type II
HL scheme [4]	$2n_u + 2n_m + 5$	4	4	N	N
ZW scheme [5]	7	10	2(+4)	Y	N
Our scheme	9	14	2(+7)	Y	Y

From Table 1, we can see that our improved scheme is not only secure but also more efficient than the scheme [4] presented by Hwang et al.. Comparing with ZW scheme [5], although the computational cost of our scheme is relatively high, our scheme overcomes the security weakness of ZW scheme. Moreover, our scheme preserves its most distinctive feature of a short public key length though based on a stronger assumption that depends on the total number of partial private key extraction queries made by the adversary.

5 Conclusion

We have presented a probably-secure certificateless encryption scheme without random oracles to correct the security drawbacks of ZW scheme. Meanwhile, our improved scheme preserves the original scheme's most distinctive features of a short public key length and a lower computational cost. Our security proofs in the standard model demonstrate that our construction is secure against Type I adversaries and Type II adversaries (including malicious KGCs) under the decisional truncated q-ABDHE assumption and the DBDH assumption, respectively.

Acknowledgments. This work is supported by National Natural Science Foundation of China (No.61170298), Natural Science Foundation of Jiangsu Province (No.BK2011101).

References

1. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
2. Al-Riyami, S., Paterson, K.: Certificateless Public Key Cryptography. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg (2003)
3. Au, M., Chen, J., Liu, J., Mu, Y., Wong, D., Yang, G.: Malicious KGC Attacks in Certificateless Cryptography. In: Proc. ASIACCS 2007, pp. 302–311. ACM press (2007)
4. Hwang, Y., Liu, J.: Certificateless public key encryption secure against malicious KGC attacks in the standard model. Journal of Universal Computer Science 463–480 (2008)
5. Zhang, G., Wang, X.: Certificateless Encryption Scheme Secure in Standard Model. Tsinghua Science and Technology 452-459 (2009)
6. Shen, L., Zhang, F., Li, S.: Cryptanalysis of a Certificateless Encryption Scheme in the Standard Model. In: 4th International Conference on Intelligent Networking and Collaborative Systems, INCoS 2012 (2012)
7. Dent, A.: A Survey of Certificateless Encryption Schemes and Security Models. Cryptology ePrint Archive, Report 2006/211 (2006)
8. Dent, A.W., Libert, B., Paterson, K.G.: Certificateless Encryption Schemes Strongly Secure in the Standard Model. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 344–359. Springer, Heidelberg (2008)

9. Al-Riyami, S., Paterson, K.: CBE from CL-PKE: A Generic Construction and Efficient Schemes. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 398–415. Springer, Heidelberg (2005)
10. Baek, J., Safavi-Naini, R., Susilo, W.: Certificateless Public Key Encryption Without Pairing. In: Zhou, J., López, J., Deng, R.H., Bao, F. (eds.) ISC 2005. LNCS, vol. 3650, pp. 134–148. Springer, Heidelberg (2005)
11. Bentahar, K., Farshim, P., Malone-Lee, J., Smart, N.: Generic construction of identity-based and certificateless KEMs. Cryptology ePrint Archive, Report 2005/058 (2005), http://eprint.iacr.org/2005/058
12. Cheng, Z., Comley, R.: Efficient Certificateless Public Key Encryption. Cryptology ePrint Archive, Report 2005/012 (2005), http://eprint.iacr.org/2005/012
13. Huang, X., Susilo, W., Mu, Y., Zhang, F.T.: On the Security of Certificateless Signature Schemes from Asiacrypt 2003. In: Desmedt, Y.G., Wang, H., Mu, Y., Li, Y. (eds.) CANS 2005. LNCS, vol. 3810, pp. 13–25. Springer, Heidelberg (2005)
14. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
15. Libert, B., Quisquater, J.-J.: On Constructing Certificateless Cryptosystems from Identity Based Encryption. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 474–490. Springer, Heidelberg (2006)
16. Huang, Q., Wong, D.S.: Generic Certificateless Encryption in the Standard Model. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC 2007. LNCS, vol. 4752, pp. 278–291. Springer, Heidelberg (2007)
17. Huang, Q., Wong, D.: Generic Certificateless Encryption Secure Against Malicious-but-Passive KGC Attacks in the Standard Model. Journal of Computer Seience and Technology, 807–826 (2010)
18. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
19. Bellare, M., Shoup, S.: Two-tier Signatures, Strongly Unforgeable Signatures, and Fiat-Shamir without Random Oracles. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 201–216. Springer, Heidelberg (2007)
20. Yum, D., Lee, P.: Generic Construction of Certificateless Signature. In: Wang, H., Pieprzyk, J., Varadharajan, V. (eds.) ACISP 2004. LNCS, vol. 3108, pp. 200–211. Springer, Heidelberg (2004)

(Strong) Multi-Designated Verifiers Signatures Secure against Rogue Key Attack

Yunmei Zhang, Man Ho Au, Guomin Yang, and Willy Susilo

Centre for Computer and Information Security Research (CCISR)
School of Computer Science and Software Engineering
University of Wollongong, Australia
yz841@uowmail.edu.au, {aau,gyang,wsusilo}@uow.edu.au

Abstract. Designated verifier signatures (DVS) allow a signer to create a signature whose validity can only be verified by a specific entity chosen by the signer. In addition, the chosen entity, known as the designated verifier, cannot convince any body that the signature is created by the signer. Multi-designated verifiers signatures (MDVS) are a natural extension of DVS in which the signer can choose multiple designated verifiers. DVS and MDVS are useful primitives in electronic voting and contract signing. In this paper, we investigate various aspects of MDVS and make two contributions. Firstly, we revisit the notion of unforgeability under rogue key attack on MDVS. In this attack scenario, a malicious designated verifier tries to forge a signature that passes through the verification of another honest designated verifier. A common counter-measure involves making the knowledge of secret key assumption (KOSK) in which an adversary is required to produce a proof-of-knowledge of the secret key. We strengthened the existing security model to capture this attack and propose a new construction that does not rely on the KOSK assumption. Secondly, we propose a generic construction of strong MDVS.

1 Introduction

Designated verifier signatures/proofs (DVS/DVP) were introduced by Jakobsson, Sako and Impagliazzo [12], and independently by Chaum [6] in 1996. A DVS scheme allows a signer Alice to convince a designated verifier Bob that Alice has endorsed the message while Bob cannot transfer this conviction to anyone else. The underlying principle of DVS is that a signature is a non-interactive proof that asserts the validity of the statement "Alice has endorsed a message" or "the signer has Bob's secret key". While Bob is convinced that Alice has endorsed the message, he cannot convince Carol as the proof could have been produced by Bob himself. In the same paper, Jakobsson *et al.* introduce the concept of strong DVS (SDVS) in which the private key of Bob is required to verify the signature. Recall that DVS itself discloses the information that the signature is produced by Alice or Bob. If an external party, Carol, is confident that Bob has not created the signature, she knows Alice has endorsed the message. An example is that the signature is captured by Carol before it reaches Bob.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 334–347, 2012.

This requirement is formalized as privacy of signer's identity in [19]. It is required that without Bob's private key, Carol cannot tell if a signature is created by Alice or another signer.

In [12], the concept of multiple verifiers has been discussed and in the rump session of Crypto'03, Desmedt [10] proposed the notion of multi-designated verifiers signatures (MDVS) as a generalization of DVS. It was later formalized in [14]. Since then, a number of MDVS constructions [5,7,8,12,14–17,20,22–24] with different features in different settings have been proposed. Interested readers may refer to [23] for a survey.

The problem of rogue key attack in DVS was first discussed in [12]. In the discussion, the goal of a malicious verifier Bob is to convince an external party Carol that the signer Alice has endorsed the message. For example, Bob can create his public key as the output of a hash function using a random number as input. Later, when Bob reveals the value of the random number, everyone will be convinced that the signatures must have been created by Alice. One of the counter-measures suggested is to require Bob to prove the knowledge of his secret key. Another type of rogue key attack specifically targeting MDVS was discussed in [21]. In this attack, a malicious verifier Carol creates her public key as a function of other honest verifiers' public keys so that she could create a signature that passes the verification of other honest verifiers. Again, the suggested counter-measure is to require the verifier to prove the knowledge of her secret key. Note that no formal model has been proposed to capture the attack. We remark that the two types of rogue key attacks are different in nature. The former is against non-transferability while the latter is against unforgeability. In this paper, our focus is on the latter.

As discussed, a counter-measure against rogue key attack in MDVS is to require the adversary to produce a proof-of-knowledge of the secret key. In practice, this implies all users would have to produce a proof-of-knowledge of the secret key to the certification authority (CA) before the CA certifies the corresponding public key. This solution requires a change in the current PKI and is regarded as costly [2]. Thus, it is desirable to design MDVS secure against rogue key attack in the plain model. In respond to this, we provide a partial solution by proposing the first MDVS scheme that is formally proven unforgeable under rogue key attack.

It is known that if we encrypt the DVS under the designated verifier's public key, the resulting scheme would be a strong DVS. Nonetheless, a subtle issue discussed in [14] prevent such generic transformation to be applicable to the case of MDVS. Specifically, the challenge is to ensure correctness of the resulting scheme since it is entirely possible for a signer to encrypt different values under different designated verifier's public key so that a signature could be regarded as valid by some of the designated verifiers only. We tackle this issue with an hybrid encryption using a simple one-way secure encryption and a symmetric encryption and show that the unforgeability under rogue key attack is preserved in our generic transformation. Specifically, we make the following contributions.

1.1 Contribution

1. We present a formal definition for MDVS that captures existential forgery under rogue key attack.
2. We propose a construction that is provably secure against rogue key attack in our model.
3. We present a generic construction of strong MDVS secure against rogue key attack.

Organization. The rest of the paper is organized as follows. In Section 2, we review the syntax of a MDVS scheme and its security definitions. We discuss the rogue key attack on MDVS, its formal definition and our proposed solution in Section 3. In Section 4, we present a generic construction of strong MDVS. We conclude our paper in Section 5.

2 Preliminary

If n is a positive integer, we use $[n]$ to denote the set $\{1, \ldots, n\}$. We review the following well-known computational assumptions.

Definition 1 (DL Assumption). *Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. The discrete logarithm assumption states that given a tuple $(g, Z) \in (\mathbb{G}, \mathbb{G})$, it is computationally infeasible to compute the value $z \in \mathbb{Z}_p$ such that $Z = g^z$.*

Definition 2 (CDH Assumption). *Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. The computational Diffie-Hellman assumption states that given a tuple $(g, g^a, g^b) \in (\mathbb{G}, \mathbb{G}, \mathbb{G})$, it is computationally infeasible to compute the value g^{ab}.*

2.1 Syntax

We adapt the definitions and security models of MDVS from various literatures [14, 15]. A MDVS scheme consists of four algorithms, namely, Setup, Gen, Sign, Verify, whose functions are enumerated below.

$\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$: On input a security parameter λ, this algorithm outputs the public parameter param for the system. Note that this algorithm is *optional* if all users could generate their key pairs without any coordination. Nonetheless, to the best of our knowledge, all existing schemes requires the users to create their keys based on some commonly known system parameters. We assume param is an implicit input to all algorithms listed below.

$(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}()$: This algorithm outputs a key pair $(\mathsf{pk}, \mathsf{sk})$ for a user (who can take the role of a signer or a designated verifier). If $(\mathsf{pk}, \mathsf{sk})$ is an output of the algorithm $\mathsf{Gen}()$, we say pk is the corresponding public key of sk (and vice versa).

$(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$: On input a message m, a secret key of a signer sk_S (whose public key is pk_S) and a set of designated verifiers' public keys $\mathcal{V}$, this algorithm outputs a signature σ, which is a designated verifier signature of m with respect to the public key pk_S.

$\texttt{valid/invalid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_V)$: On input a public key pk_S, a message m, a signature σ with a set of designated verifiers' public keys $\mathcal{V}$ and a private key sk_V such that the corresponding public key $\mathsf{pk}_V \in \mathcal{V}$, this algorithm verifies the signature and outputs `valid/invalid`.

A MDVS scheme must possess *Correctness*, *Unforgeability* and *Source-Hiding*, to be reviewed below.

Correctness. For any security parameter λ and $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$, $(\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{Gen}()$ and $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ such that $(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}) \leftarrow \mathsf{Gen}()$ for $i \in [n]$. For any message m, if $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$, then $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_{V_i})$ for all $i \in [n]$. Furthermore, for any values σ, $\mathcal{V}$, m, pk_S, if there exists a private key sk_V such that its corresponding public key $\mathsf{pk}_V \in \mathcal{V}$ and that $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_V)$, then for any private key $\mathsf{sk}_{V'}$, it holds that $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V}, m, \mathsf{sk}_{V'})$ if the corresponding public key $\mathsf{pk}_{V'} \in \mathcal{V}$.

Unforgeability. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally captures the requirement of *Unforgeability*.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and subsequently $\mathsf{Gen}()$ to obtain $(\mathsf{param}, (\mathsf{pk}_S, \mathsf{sk}_S), \{(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i})\}_{i\in[n]})$. Denote the set $\{\mathsf{pk}_{V_i}\}_{i\in[n]}$ by $\mathcal{V}$. $(\mathsf{param}, \mathsf{pk}_S, \mathcal{V})$ is given to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to make the following queries:

- Corruption Query. $\mathcal{A}$ submits a public key $\mathsf{pk}_V \in \mathcal{V}$ and receives sk_V.
- Signature Query. $\mathcal{A}$ submits a message m and receives $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$.

Output $\mathcal{A}$ submits (σ^*, m^*) and wins if and only if

1. There exists a public key $\mathsf{pk}_{V^*} \in \mathcal{V}$ such that $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma^*, \mathcal{V}, m^*, \mathsf{sk}_{V^*})$.
2. $\mathcal{A}$ has not submitted a Signature Query with input m^*.
3. There exists a public key $\mathsf{pk}_V \in \mathcal{V}$ such that $\mathcal{A}$ has not submitted a Corruption Query as input.

Definition 3 (Unforgeability). *A MDVS scheme is unforgeable if no PPT adversary wins the above game with non-negligible probability.*

As stated in [14], the adversary is not given an oracle for signature verification as he can verify any signatures by corrupting some of the verifiers.

Source Hiding. It means that given a message m and a signature $(\sigma, \mathcal{V})$, it is infeasible to determine who from the original signer or the designated verifiers all together created the signature, even if all the secret keys are known. The formal definition is adapted from Definition 3 of [11] for normal DVS into that for MDVS.

Definition 4 (Source Hiding). *A MDVS scheme is* source hiding *if there exists a PPT simulation algorithm* Sim *that on input a public key* pk_S*, a set of key pairs* $(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i})_{i\in[n]}$ *and a message* m*, outputs a tuple* $(\sigma, \mathcal{V})$ *(such that* $\mathcal{V} = \{\mathsf{pk}_{V_i}\}_{i\in[n]}$*) that is indistinguishale to* $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m)$ *(where* sk_S *is the corresponding private key of* pk_S*). In other words, for all PPT algorithmn* $\mathcal{D}$*, for any security parameter* λ*,* $\mathsf{param} \leftarrow \mathsf{Setup}(1^\lambda)$*,* $(\mathsf{pk}_S, \mathsf{sk}_S) \leftarrow \mathsf{Gen}()$*,* $\{(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}) \leftarrow \mathsf{Gen}()\}_{i\in[n]}$ *and any message* m*, it holds that:*

$$\left| Pr\left[\begin{array}{r} (\sigma_0, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_S, \mathcal{V}, m) \\ (\sigma_1, \mathcal{V}) \leftarrow \mathsf{Sim}(\mathsf{pk}_S, \{\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}\}_{i\in[n]}, m) \\ b \in_R \{0,1\} \\ b' \leftarrow \mathcal{D}(\sigma_b, \mathsf{pk}_S, \mathsf{sk}_S, \{\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i}\}_{i\in[n]}, m) \end{array} : b = b' \right] - 1/2 \right| = \mathsf{negl}(\lambda)$$

where $\mathsf{negl}(\lambda)$ *represents a negligible function in* λ*. A function* $\mathsf{negl}(\lambda)$ *is said to be negligible in* λ *if for all polynomial* $q(\cdot)$*, there exists a value* k_0 *such that for every* $\lambda > k_0$*,* $\mathsf{negl}(\lambda) < 1/q(\lambda)$*.*

2.2 Strong Multi-Designated Verifiers Signatures

Strong Multi-Designated Verifiers Signatures. It is desirable in many scenarios that, besides the signer and the verifier, a third party cannot tell if a signature for the verifier is created by that particular signer or by someone else. This concept appeared in [12] and is formally defined as privacy of signer's identity (PSI) in [13]. This applies to the case of multiple designated verifiers and the property PSI for MDVS is defined in [14].

Privacy of Signer's Identity. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally captures the requirement of *PSI*.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and subsequently $\mathsf{Gen}()$ to obtain $(\mathsf{param}, (\mathsf{pk}_{S_0}, \mathsf{sk}_{S_0}), (\mathsf{pk}_{S_1}, \mathsf{sk}_{S_1}), \{(\mathsf{pk}_{V_i}, \mathsf{sk}_{V_i})\}_{i\in[n]})$. Denote the set $\{\mathsf{pk}_{V_i}\}_{i\in[n]}$ by $\mathcal{V}$. $(\mathsf{param}, \mathsf{pk}_{S_0}, \mathsf{pk}_{S_1}, \mathcal{V})$ is given to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to make the following queries:

- Verification Query. $\mathcal{A}$ submits $(m, \sigma, \mathcal{V}, \mathsf{pk}_{S_c} : c \in \{0,1\}, V \in \mathcal{V})$ and receives $\mathtt{valid/invalid} \leftarrow \mathsf{Verify}(\mathsf{pk}_{S_c}, \sigma, \mathcal{V} \cup \mathsf{pk}_V, m, \mathsf{sk}_V)$.
- Signature Query. $\mathcal{A}$ submits a message m, a bit b and receives $(\sigma, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_{S_b}, \mathcal{V}, m)$.

Challenge At some point $\mathcal{A}$ submits a message m^*. $\mathcal{C}$ flips a fair coin b and returns $(\sigma^*, \mathcal{V}) \leftarrow \mathsf{Sign}(\mathsf{sk}_{S_b}, \mathcal{V}, m)$.

Query $\mathcal{A}$ continues to make verification and signature queries.

Output $\mathcal{A}$ submits a bit b' and wins if and only if $b' = b$.

$\mathcal{A}$'s advantage in the game PSI is defined as the probability that $\mathcal{A}$ wins the game minus 1/2.

Definition 5 (Privacy of signer's identity). *A MDVS scheme is said to possess privacy of signer's identity if no PPT adversary has non-negligible advantage in game PSI.*

A strong MDVS scheme is a MDVS scheme that possesses privacy of signer's identity.

3 Rouge Key Attack in MDVS and Its Solution

We first review the generic construction of MDVS from discrete logarithm-based ring signatures [14]. In the next subsection, we describe how a malicious designated verifier could launch a rogue key attack to make an honest verifier into accepting a forged signature. We stress that this attack is outside the original security model and does not imply the scheme is insecure. Rather, we would like to show that a signature that passes the verification of a particular honest designated verifier could have been created by a real signer or some other malicious verifiers. Finally, we propose a fix.

3.1 Generic Construction of MDVS [14]

The generic construction utilizes ring signatures as building blocks and requires that all the keys are discrete logarithm-based. Readers are referred to [18] for the formal definition of a ring signature scheme. Roughly speaking, a ring signature is a signature created from one of the possible signers in a set of signers (often called a ring of signers). The ring of signers are created in an ad-hoc manner by the actual signer. The formation is spontaneous in that the members can be completely unaware of being conscripted into the ring. In the generic construction of MDVS, ring signatures supporting a ring size of 2 is required.

- Setup. This is equivalent to the parameter generation of the ring signature scheme (if any).
- Gen. This is equivalent to the key generation of the ring signature scheme. The generic construction requires the key of the ring signature to be of the form (g^x, x) where g is included in the parameter, x is the signing key and g^x is the corresponding public key.
- Sign. Let the signer's key pair be (g^{x_S}, x_S) and the set of designated verifiers' key pairs be $\{(g^{x_{V_i}}, x_{V_i})\}$ for $i = 1$ to n. The signer computes $g^{X_V} = \prod_{i\in[n]} g^{x_{V_i}}$. Next, the signer creates a ring signature on message m on the ring $\{g^{x_S}, g^{x_V}\}$ using the secret key x_S. Denote the output as σ. This value, together with the set $\{g^{x_{V_i}}\}_{i\in[n]}$, is outputted as the multi-designated verifier signature.
- Verify. To verify the signature $(\sigma, \{g^{x_{V_i}}\}_{i\in[n]})$ on message m, a verifier computes $g^{X_V} = \prod_{i\in[n]} g^{x_{V_i}}$. Then it employs the verification algorithm of the ring signature scheme on the ring $\{g^{X_S}, g^{X_V}\}$.

The *unforgeability* property comes from the fact that to create a ring signature on the ring $\{g^{x_S}, g^{x_V}\}$, one needs to know x_S or x_V. Since the adversary does not know x_S or x_V[1], forging a signature implies breaking the unforgeability of the underlying ring signature scheme. On the other hand, the *source hiding* property comes from the fact that if all secret keys of the verifiers are known, one can construct a PPT Sim which computes $x_V = \sum_{i\in[n]} x_{V_i}$ and uses it to

[1] Since the adversary cannot corrupt all the verifiers, it does not know the value x_V, which is equal to $\sum_{i\in[n]} x_{V_i}$.

create a ring signature on behalf of the ring $\{g^{x_S}, g^{x_V}\}$. Due to the anonymity of ring signature, no PPT algorithm can distinguish a signature created by the real signer using x_S or by Sim using x_V.

3.2 Rouge Key Attack and Its Defence

Existential Forgery under Rouge Key Attack. Rouge key attack against a concrete scheme in [14] has been discussed in [21]. Here we extend the attack to the generic construction of [14]. Suppose an adversary's goal is to convince an honest designated verifier into accepting a forged signature. Let g^{x_S}, $g^{x_{V'}}$ be the public keys of the targeted signer and designated verifier respectively. To cheat the verifier, the adversary randomly generates a value x_A and crafts a mal-formed public key $K = g^{x_A}/g^{x_{V'}}$. Next, the adversary computes $g^{x_V} = Kg^{x_{V'}} = g^{x_A}$. Since the adversary is in possession of x_A, he can create a ring signature on the ring $\{g^{x_S}, g^{x_V}\}$. He outputs the signature, together with the set of designated verifiers as $\{g^{x_{V'}}, K\}$. Consequently, the designated verifier would accept a forged signature created by the adversary instead of the signer. We denote attack of this kind as forgery against rogue key attack (RKA).

A Proposed Fix. The problem comes from the extra power given to the adversary to create malformed public key. The fix suggest in [21] is to require the certification authority to check the validity of the public key before issuing a digital certificate. In terms of modelling, this implies the stronger certified key model in which the users are required to conduct a proof-of-knowledge of his secret key to the CA. As argue in [2], this requires modification of the client and CA functioning software. We propose another way that could withstand this attack in the plain model based on a technique used in multisignautres [2] and batch verification of digital signatures [1]. In a nutshell, g^{x_V} is defined to be $\prod_{i\in[n]}(g^{x_{V_i}})^{h_i}$, where $h_i = H(g^{x_S}, g^{x_{V_1}}, \ldots, g^{x_{V_n}}, m, i)$ for a hash function H which shall be modelled as a random oracle. Observe that with this modification, the value x_V can still be computed if all the values x_{V_i} are known. On the other hand, if one of the secret keys, say x_{V_i}, is unknown, the value x_V cannot be computed since the probability of "canceling" $g^{x_{V_i}}$ in the computation of g^{x_V} is negligible assuming the values h_i are randomly distributed and are only known after the value of the public keys are chosen.

3.3 Formal Security Definition for Unforgeability under Rogue Key Attack

To formally assert the security of our proposed solution, we define a security model which intends to capture attack of this kind.[2] We believe a verification query with the target verifier may be of use to the adversary since the adversary might try to submit mal-formed signatures to learn information about the target verifier's verification procedure.

[2] While rogue key attack on MDVS is discussed in [21], no formal security model has been proposed to capture such an attack.

Unforgeability against Rogue Key Attack. The following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$ formally captures the requirement of *UF-RKA*

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and subsequently $\mathsf{Gen}()$ to obtain $(\mathsf{param}, (\mathsf{pk}_S, \mathsf{sk}_S), (\mathsf{pk}_V, \mathsf{sk}_V))$. $(\mathsf{param}, \mathsf{pk}_S, \mathsf{pk}_V)$ is given to $\mathcal{A}$.

Query $\mathcal{A}$ is allowed to make the following queries:

- Verification Query. $\mathcal{A}$ submits a set of public keys $\mathcal{V}$, a signature $(\sigma, \mathcal{V} \cup \mathsf{pk}_V)$, a message m and receives $\texttt{valid/invalid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma, \mathcal{V} \cup \mathsf{pk}_V, m, \mathsf{sk}_V)$.
- Signature Query. $\mathcal{A}$ submits a message m, a set of public keys $\mathcal{V}$ and receives $(\sigma, \mathcal{V}) \leftarrow (\mathsf{sk}_S, \mathcal{V}, m)$. Note that $\mathcal{A}$ can submit an arbitrary set of verifiers of his choice (even a set without pk_V).

Output $\mathcal{A}$ submits (σ^*, m^*) and a set of public keys $\mathcal{V}^*$ and wins if and only if

1. $\texttt{valid} \leftarrow \mathsf{Verify}(\mathsf{pk}_S, \sigma^*, \mathcal{V}^* \cup \mathsf{pk}_V, m^*, \mathsf{sk}_V)$.
2. $\mathcal{A}$ has not submitted a Signature Query with input $(m^*, \mathcal{V}^* \cup \mathsf{pk}_V)$.

Definition 6 (UF-RKA). *A MDVS scheme is unforgeable under rogue key attack if no PPT adversary wins the above game with non-negligible probability.*

We believe UF-RKA for MDVS is a stronger notion compared with the notion *Unforgeability*.

3.4 A Concrete Construction

We present a concrete MDVS scheme from a commonly used two-party ring signature following the generic construction together with our proposed fix.

- Setup. Let $\mathbb{G} = \langle g \rangle$ be a cyclic group of prime order p. Output param as $(\mathbb{G}, p, g)$.
- Gen. Choose a hash function $H : \{0,1\}^* \to \mathbb{Z}_p$ which will be modelled as a random oracle[3]. Randomly generate $x \in_R \mathbb{Z}_p$, compute g^x. Output pk as (g^x, H) and sk as x.
- Sign. On input the signer's key pair $(\mathsf{pk}_S, \mathsf{sk}_S)$, a set of designated verifier's public keys $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ and a message m, parse pk_S as (Y_S, H_S), sk_S as x_S, pk_{V_i} as (Y_{V_i}, H_{V_i}). Compute $Y = \prod_{i \in [n]} Y_{V_i}^{h_i}$ where $h_i = H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)$.
  1. Randomly generate $r, c_2, z_2 \in \mathbb{Z}_p$, compute $T_1 = g^r$, $T_2 = Y^{c_2} g^{z_2}$.
  2. Compute $c = H_S(T_1, T_2, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y, m)$ and $c_1 = c - c_2$.
  3. Compute $z_1 = r - c_1 x_S$.

  Output the signature as $(c_1, c_2, z_1, z_2, \mathcal{V})$. Note that (c_1, c_2, z_1, z_2) is a ring signature on message m with respective to ring $\{Y_S, Y\}$.

[3] We abuse the notation and assume a full domain hash. In the following when we write $c = H(X, Y)$ where X and Y may be elements from different domains, we assume a suitable encoding scheme is employed to convert X, Y into a bit-string.

– We remark that the above steps constitute a standard signature proof-of-knowledge of 1-out-of-2 discrete logarithms (which can be viewed as a two-party ring signature). This can be presented as follows using the Camenisch and Stadler notation [4].

$$\mathrm{SPK}\{(\alpha) : Y_S = g^{\alpha} \vee Y = g^{\alpha}\}(m)$$

– Verify. To verify the signature $(c_1, c_2, z_1, z_2, \{\mathsf{pk}_{V_i}\}_{i\in[n]})$ on message m, a verifier parses pk_{V_i} as (Y_{V_i}, H_{V_i}) and computes $Y = \prod_{i\in[n]} Y_{V_i}^{H_{V_i}(\mathsf{pk}_S, \mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}, m)}$. Output `valid` if and only if

$$c_1 + c_2 = H_S\big(Y_S^{c_1} g^{z_1}, Y^{c_2} g^{z_2}, \mathsf{pk}_S, \mathsf{pk}_{V_1}, h_1, \ldots, \mathsf{pk}_{V_n}, h_n, Y, m\big)$$

and `invalid` otherwise.

Regarding the security of our concrete construction, we have the following theorem, whose proof shall appear in the full version of the paper due to page limitation.

Theorem 1. *Our concrete construction is secure under the discrete logarithm assumption in the random oracle model. Specifically, it satisfies*

– *definition 3 under the discrete logarithm assumption in the random oracle model;*
– *definition 4 unconditionally;*
– *definition 6 under the discrete logarithm assumption in the random oracle model.*

4 Generic Strong MDVS

Strong DVS can be constructed from DVS via encrypting the signature under the designated verifier's public key. However, the intuitive solution of encrypting the signature under each designated verifier's public key in the case of multiple designated verifiers is not satisfactory. As discussed in [14], this intuitive solution creates a subtle issue in correctness. Specifically, if some of the encryptions are not executed properly, the signer could create an "invalid" signature that would be regarded as valid by some verifiers.

4.1 Overview of Our Generic Construction

To tackle this challenge, we observe that it is straightforward to use a verifiable encryption [3] which allows the signer to create a proof that all ciphertext decrypts to the same value. By verifying the proof, all verifiers are assured that all the verifiers obtains the same value for signature verification. This solution is, however, expensive. Looking at an abstract level, the goal of this encryption is to ensure all verifiers obtains the same value via decryption. This can be achieved,

perhaps somewhat interestingly, using a very weak one-way encryption with an explicit "IND-CPA" attack. Denote such an encryption scheme as $\mathcal{WE}$. That is, given a message k, anyone can check if the ciphertext C decrypts to it. It is easy for a verifier to check locally if all the encryptions of the designated verifier signature are properly done.

This creates another problem. Since $\mathcal{WE}$ is only one-way secure, the ciphertext might leak information about the signature being encrypted and thus privacy of signer's identity is not guaranteed. Thus, we employ a hybrid approach. $\mathcal{WE}$ is used to encrypt a symmetric key k under all the designated verifiers public keys into ciphertexts $C_1, \ldots, C_n$. The ordinary MDVS is encrypted with a symmetric key encryption $\mathcal{SE}$ with key k. As long as the key k cannot be recovered from the ciphertext C_i's, no information about the MDVS can be learnt as long as the symmetric encryption $\mathcal{SE}$ is secure. Looking ahead, we assume $\mathcal{SE}$ to be an idealized cipher for the ease of security analysis. This means that our generic construction is secure in the ideal cipher model, which is equivalent to the random oracle model due to the result of [9].

4.2 Building Block of Our Generic Construction

While conceptually simple, two properties regarding $\mathcal{WE}$ are needed. The first one is an efficient and explicit "IND-CPA" attack. The second one is an efficient and explicit malleability attack which allows anyone to transform a ciphertext C under public key Y into another ciphertext C' under public key Y' so that they are encrypting the same message. The malleability attack on $\mathcal{WE}$ is needed in the proof of security for multiple designated verifiers.

Below we define the requirement of the weakly secure encryption $\mathcal{WE}$ as follows.

- $\mathsf{param}_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Setup}(1^\lambda)$: On input a security parameter λ, this algorithm outputs the public parameter $\mathsf{param}_{\mathcal{WE}}$ for the system. We assume $\mathsf{param}_{\mathcal{WE}}$ is an implicit input to all algorithms listed below.
- $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk}) \leftarrow \mathcal{WE}.\mathsf{Gen}()$: This algorithm outputs a key pair $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk})$.
- $C_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}, m)$: On input a message m and a public key of the receiver $\mathcal{WE}.\mathsf{pk}$, this algorithm outputs the ciphertext $C_{\mathcal{WE}}$.
- $m \leftarrow \mathcal{WE}.\mathsf{Dec}(\mathcal{WE}.\mathsf{sk}, C_{\mathcal{WE}})$: On input a secret key $\mathcal{WE}.\mathsf{sk}$, a ciphertext $C_{\mathcal{WE}}$, this algorithm outputs the plaintext m.
- $0/1 \leftarrow \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}}, m)$: This is an attack on indistinguishability of ciphertext. On input a public key $\mathcal{WE}.\mathsf{pk}$, a ciphertext $C_{\mathcal{WE}}$ and a plaintext m, output 1 if and only if $m = \mathcal{WE}.\mathsf{Dec}(\mathcal{WE}.\mathsf{sk}, C_{\mathcal{WE}})$, where $\mathcal{WE}.\mathsf{sk}$ is the corresponding private key of $\mathcal{WE}.\mathsf{pk}$ and 0 otherwise. Note that $\mathcal{WE}.\mathsf{sk}$ is not an input to this algorithm.
- $(C'_{\mathcal{WE}}, \mathcal{WE}.\mathsf{pk}') \leftarrow \mathcal{WE}.\mathsf{mAtk}(\mathcal{WE}.\mathsf{pk}, C_{\mathcal{WE}})$: This is an attack on malleability of ciphertext. On input a public key $\mathcal{WE}.\mathsf{pk}$, a ciphertext $C_{\mathcal{WE}}$, output $C'_{\mathcal{WE}}$, $\mathcal{WE}.\mathsf{pk}'$ such that the distribution of $C'_{\mathcal{WE}}$ is indistinguishable to that of $\mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}', \mathcal{WE}.\mathsf{Dec}(\mathcal{WE}.\mathsf{sk}, C_{\mathcal{WE}}))$. Note that the algorithm does not output the corresponding secret key for $\mathcal{WE}.\mathsf{pk}'$.

We require the one-way security of $\mathcal{WE}$, which is formally defined as the following game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

Setup $\mathcal{C}$ invokes $\mathsf{Setup}(1^\lambda)$ and subsequently $\mathsf{Gen}()$ to obtain ($\mathsf{param}_{\mathcal{WE}}$, $\mathcal{WE}.\mathsf{pk}$, $\mathcal{WE}.\mathsf{sk}$).

Challenge $\mathcal{C}$ picks a random message m, compute $C_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}, m)$. ($\mathsf{param}_{\mathcal{WE}}$, $\mathcal{WE}.\mathsf{pk}$, $C_{\mathcal{WE}}$) is given to $\mathcal{A}$.

Output $\mathcal{A}$ outputs m' and win if and only if $m = m'$.

$\mathcal{WE}$ is one-way secure if no PPT adversary $\mathcal{A}$ wins the above game with non-negligible probability.

We propose a construction of $\mathcal{WE}$ based on the Elgamal encryption in a cyclic group equipped with a bilinear map.

- $\mathcal{WE}.\mathsf{Setup}(1^\lambda)$: Generate a pair of groups $\mathbb{G}, \mathbb{G}_T$ of the same prime order p of λ-bit and a bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. Let g be a generator of $\mathbb{G}$. Set $\mathsf{param}_{\mathcal{WE}} = (\mathbb{G}, \mathbb{G}_T, p, g, \hat{e})$.
- $\mathcal{WE}.\mathsf{Gen}()$: Randomly pick $u \in_R \mathbb{Z}_p$, compute $U = g^u$. Set $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk}) = (U, u)$.
- $\mathcal{WE}.\mathsf{Enc}(U, m)$: On input a message $m \in \mathbb{G}$, randomly generate $r \in_R \mathbb{Z}_p$, output $C_{\mathcal{WE}} = (C, D)$ as (mU^r, g^r).
- $\mathcal{WE}.\mathsf{Dec}(u, (C, D))$: Output C/D^u.
- $\mathcal{WE}.\mathsf{iAtk}(U, (C, D), m)$: Output 1 if and only if

$$\hat{e}(C/m, g) = (D, U)$$

  and 0 otherwise.
- $\mathcal{WE}.\mathsf{mAtk}(U, (C, D))$: Randomly pick $e, f \in_R \mathbb{Z}_p$, compute $U' = Ug^e$. Compute $\tilde{C} = CD^e$, $C' = \tilde{C}U'^f$, $D' = Dg^f$. Output $\big((C', D'), U'\big)$.

Note that if $U = g^u$, $C = mU^r$, $D = g^r$, it is easy to see that $C' = m(Ug^e)^{r+f}$, $D' = g^{r+f}$ and $U' = Ug^e$. Thus, (C', D') is encrypting the message m under the public key U' with the correct distribution.

Next, we show that our construction of $\mathcal{WE}$ is one-way secure under the computational Diffie-Hellman assumption.

Proof. Suppose there exists an adversary $\mathcal{A}$ that can win the game one-way security, we show how to construction an algorithm $\mathcal{S}$ that solves the CDH problem in a group equipped with a bilinear map. $\mathcal{S}$ is given $(\mathbb{G}, \mathbb{G}_T, \hat{e}, p, g, g^a, g^b)$ and its goal is to output g^{ab}.

$\mathcal{S}$ randomly picks a value C, gives $\mathsf{param}_{\mathcal{WE}} = (\mathbb{G}, \mathbb{G}_T, p, g, \hat{e})$, $U = g^a$, $(C, D) = (C, g^b)$ to $\mathcal{A}$. Note that this implicit set the message being encrypted as $m = C/g^{ab}$. $\mathcal{A}$ returns with a value m'. $\mathcal{S}$ computes C/m' and outputs it as the solution to the CDH problem. □

4.3 Our Generic Construction of Strong MDVS

We present our generic construction of Strong MDVS. Let $\mathcal{MS} = (\mathcal{MS}.\mathsf{Setup}, \mathcal{MS}.\mathsf{Gen}, \mathcal{MS}.\mathsf{Sign}, \mathcal{MS}.\mathsf{Verify})$ be a secure MDVS scheme. Let $\mathcal{WE} = (\mathcal{WE}.\mathsf{Setup}, \mathcal{WE}.\mathsf{Gen}, \mathcal{WE}.\mathsf{Enc}, \mathcal{WE}.\mathsf{Dec}, \mathcal{WE}.\mathsf{iAtk}, \mathcal{WE}.\mathsf{mAtk})$ be a one-way secure encryption. Let H be a hash function and $\mathcal{SE}$ be a symmetric key encryption. We use $\mathcal{SE}.\mathsf{Enc}_k$ and $\mathcal{SE}.\mathsf{Dec}_k$ to denote encryption and decryption operation of $\mathcal{SE}$ using key k. H, $\mathcal{SE}$ will be modelled as a random oracle and an ideal cipher respectively. We show how to construct a strong MDVS scheme (Setup, Gen, Sign, Verify) as follows.

- Setup. On input security parameter 1^λ, invoke $\mathsf{param}_{\mathcal{MS}} \leftarrow \mathcal{MS}.\mathsf{Setup}(1^\lambda)$ and $\mathsf{param}_{\mathcal{WE}} \leftarrow \mathcal{WE}.\mathsf{Setup}(1^\lambda)$, specify a weak encryption $\mathcal{WE}$, a hash function H and a symmetric cipher $\mathcal{SE}$. Set $\mathsf{param} = (\mathsf{param}_{\mathcal{MS}}, \mathsf{param}_{\mathcal{WE}}, H, \mathcal{SE})$.
- Gen. Invoke $(\mathcal{MS}.\mathsf{pk}, \mathcal{MS}.\mathsf{sk}) \leftarrow \mathcal{MS}.\mathsf{Gen}()$, $(\mathcal{WE}.\mathsf{pk}, \mathcal{WE}.\mathsf{sk}) \leftarrow \mathcal{WE}.\mathsf{Gen}()$. Output $\mathsf{pk} = (\mathcal{MS}.\mathsf{pk}, \mathcal{WE}.\mathsf{pk})$ and $\mathsf{sk} = (\mathcal{MS}.\mathsf{sk}, \mathcal{WE}.\mathsf{sk})$.
- Sign. Let $\mathsf{pk}_S = (\mathcal{MS}.\mathsf{pk}_S, \mathcal{WE}.\mathsf{pk}_S)$ and $\mathsf{sk}_S = (\mathcal{MS}.\mathsf{sk}_S, \mathcal{WE}.\mathsf{sk}_S)$ be the key pair of the signer. Let m be the message to be signed. Parse the set of verifiers to be $\mathcal{V} = \{\mathsf{pk}_{V_1}, \ldots, \mathsf{pk}_{V_n}\}$ such that $\mathsf{pk}_{V_i} = (\mathcal{MS}.\mathsf{pk}_i, \mathcal{WE}.\mathsf{pk}_i)$. Denote by $\mathcal{V}_{\mathcal{MS}}$ the set $\{\mathcal{MS}.\mathsf{pk}_1, \ldots, \mathcal{MS}.\mathsf{pk}_n\}$.
  The signer randomly picks $k \in_R \{0,1\}^\lambda$. For $i = 1$ to n, compute
  $$C_i = \mathcal{WE}.\mathsf{Enc}(\mathcal{WE}.\mathsf{pk}_i, k)$$
  Next, compute $\tau = H(\mathsf{pk}_S, \mathsf{pk}_{V_1}, C_1, \ldots, \mathsf{pk}_{V_n}, C_n, m)$. Invoke $(\sigma_{\mathcal{MS}}, \mathcal{V}_{\mathcal{MS}}) \leftarrow \mathsf{Sign}(\mathcal{MS}.\mathsf{sk}_S, \mathcal{V}_{\mathcal{MS}}, m||\tau)$. Invoke $E = \mathcal{SE}.\mathsf{Enc}_k(\sigma_{\mathcal{MS}}||\tau||\mathsf{pk}_S)$.
  Output the signature as $(E, \mathcal{V}, \{C_i\}_{i\in[n]})$.
- Verify. To verify a signature $(E, \mathcal{V}, \{C_i\}_{i\in[n]})$ on message m, a verifier V parses pk_{V_i} as $(\mathcal{MS}.\mathsf{pk}_i, \mathcal{WE}.\mathsf{pk}_i)$ for all $\mathsf{pk}_{V_i} \in \mathcal{V}$ and uses his secret key $(\mathcal{MS}.\mathsf{sk}_V, \mathcal{WE}.\mathsf{sk}_V)$ as follows.
  - Locate the index i such that $\mathsf{pk}_V = \mathsf{pk}_{V_i}$. Use his secret key to compute $k = \mathcal{VE}.\mathsf{Dec}(C_i, \mathcal{WE}.\mathsf{sk}_V)$.
  - For all $j \in [n]\backslash\{i\}$, check if $1 = \mathcal{WE}.\mathsf{iAtk}(\mathcal{WE}.\mathsf{pk}_i, C_i, k)$. Output `invalid` if any of the check outputs 0.
  - Compute $\sigma_{\mathcal{MS}}$, τ, pk_S by $\mathcal{SE}.\mathsf{Dec}_k(E)$.
  - Output `invalid` if $\tau \neq H(\mathsf{pk}_S, \mathsf{pk}_{V_1}, C_1, \ldots, \mathsf{pk}_{V_n}, C_n, m)$.
  - Parse pk_S as $(\mathcal{MS}.\mathsf{pk}_S, \mathcal{WE}.\mathsf{pk}_S)$.
  - Invoke `valid`/ `invalid` $\leftarrow \mathcal{MS}.\mathsf{Verify}(\mathcal{MS}.\mathsf{pk}_S, \sigma_{\mathcal{MS}}, \{\mathcal{MS}.\mathsf{pk}_1, \ldots, \mathcal{MS}.\mathsf{pk}_n\}, m||\tau, \mathcal{MS}.\mathsf{sk}_V)$.

Regarding the security of our generic construction, we have the following theorem, whose proof shall appear in the full version of the paper due to page limitation.

Theorem 2. *Our generic construction satisfies definition d if the underlying MDVS scheme $\mathcal{MS}$ satisfies definitions d for $d \in \{3, 4, 6\}$. Furthermore, our generic construction satisfies definition 5 if $\mathcal{WE}$ is one-way secure in the random oracle model.*

5 Conclusion

In this paper, we formalized the security notion unforgeability under rogue key attack for MDVS. We proposed an efficient construction that is provably secure in the proposed model. In addition, we present a generic transformation that converts any secure MDVS scheme into a strong MDVS scheme. We leave the construction of constant size strong MDVS scheme secure under our definitions as an open problem.

References

1. Bellare, M., Garay, J.A., Rabin, T.: Fast Batch Verification for Modular Exponentiation and Digital Signatures. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 236–250. Springer, Heidelberg (1998)
2. Bellare, M., Neven, G.: Multi-Signatures in the Plain Public-Key Model and a General Forking Lemma. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) ACM Conference on Computer and Communications Security, pp. 390–399. ACM (2006)
3. Camenisch, J., Shoup, V.: Practical Verifiable Encryption and Decryption of Discrete Logarithms. In: Boneh, D. (ed.) CRYPTO 2003. LNCS, vol. 2729, pp. 126–144. Springer, Heidelberg (2003)
4. Camenisch, J., Stadler, M.: Efficient Group Signature Schemes for Large Groups. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 410–424. Springer, Heidelberg (1997)
5. Chang, T.Y.: An ID-Based Multi-Signer Universal Designated Multi-Verifier Signature Scheme. Inf. Comput. 209(7), 1007–1015 (2011)
6. Chaum, D.: Private Signature and Proof Systems, US Patent 5,493,614 (1996)
7. Chow, S.S.M.: Identity-Based Strong Multi-Designated Verifiers Signatures. In: Atzeni, A.S., Lioy, A. (eds.) EuroPKI 2006. LNCS, vol. 4043, pp. 257–259. Springer, Heidelberg (2006)
8. Chow, S.S.M.: Multi-Designated Verifiers Signatures Revisited. I. J. Network Security 7(3), 348–357 (2008)
9. Coron, J.-S., Patarin, J., Seurin, Y.: The Random Oracle Model and the Ideal Cipher Model Are Equivalent. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 1–20. Springer, Heidelberg (2008)
10. Desmedt, Y.: Verifier-Designated Signatures. In: CRYPTO Rump Session (2003)
11. Huang, Q., Yang, G., Wong, D.S., Susilo, W.: Efficient Strong Designated Verifier Signature Schemes without Random Oracle or with Non-Delegatability. Int. J. Inf. Sec. 10(6), 373–385 (2011)
12. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
13. Laguillaumie, F., Vergnaud, D.: Designated Verifier Signatures: Anonymity and Efficient Construction from *Any* Bilinear Map. In: Blundo, C., Cimato, S. (eds.) SCN 2004. LNCS, vol. 3352, pp. 105–119. Springer, Heidelberg (2005)
14. Laguillaumie, F., Vergnaud, D.: Multi-Designated Verifiers Signatures. In: López, J., Qing, S., Okamoto, E. (eds.) ICICS 2004. LNCS, vol. 3269, pp. 495–507. Springer, Heidelberg (2004)

15. Laguillaumie, F., Vergnaud, D.: Multi-Designated Verifiers Signatures: Anonymity without Encryption. Inf. Process. Lett. 102(2-3), 127–132 (2007)
16. Li, Y., Susilo, W., Mu, Y., Pei, D.: Designated Verifier Signature: Definition, Framework and New Constructions. In: Indulska, J., Ma, J., Yang, L.T., Ungerer, T., Cao, J. (eds.) UIC 2007. LNCS, vol. 4611, pp. 1191–1200. Springer, Heidelberg (2007)
17. Ng, C.Y., Susilo, W., Mu, Y.: Universal Designated Multi Verifier Signature Schemes. In: ICPADS (2), pp. 305–309. IEEE Computer Society (2005)
18. Rivest, R.L., Shamir, A., Tauman, Y.: How to Leak a Secret. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 552–565. Springer, Heidelberg (2001)
19. Saeednia, S., Kremer, S., Markowitch, O.: An Efficient Strong Designated Verifier Signature Scheme. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 40–54. Springer, Heidelberg (2004)
20. Shailaja, G., Kumar, K.P., Saxena, A.: Universal Designated Multi Verifier Signature without Random Oracles. In: Mohanty, S.P., Sahoo, A. (eds.) ICIT, pp. 168–171. IEEE Computer Society (2006)
21. Shim, K.-A.: Rogue-key Attacks on the Multi-designated Verifiers Signature Scheme. Inf. Process. Lett. 107(2), 83–86 (2008)
22. Tian, H.: A New Strong Multiple Designated Verifiers Signature for Broadcast Propagation. In: Xhafa, F., Barolli, L., Köppen, M. (eds.) INCoS, pp. 268–274. IEEE (2011)
23. Tian, H.: A New Strong Multiple Designated Verifiers Signature. IJGUC 3(1), 1–11 (2012)
24. Vergnaud, D.: New Extensions of Pairing-based Signatures into Universal (Multi) Designated Verifier Signatures. CoRR, abs/0802.1076 (2008)

Direct CCA Secure Identity-Based Broadcast Encryption

Leyou Zhang[1], Qing Wu[2], and Yupu Hu[3]

[1] Department of Mathematics, School of Science, Xidian University, Xi'an, 710071, China
[2] School of Automation, Xi'an University of Posts and Telecommunications, Xi'an, 710121, China
[3] Key Laboratory of Computer Networks and Information Security, Xidian University, Xi'an, 710071, China
leyouzhang77@yahoo.com.cn

Abstract. In the previous works, the general transformation methods from a CPA(chosen-plaintext attacks) secure scheme to a CCA(chosen-ciphertext attacks) secure scheme are the hierarchical identity-based encryption, one-time signature and MAC. These folklore construction methods lead to the CCA secure schemes that are somewhat inefficient in the real life. In this paper, a new direct chosen-ciphertext technique is introduced and a practical identity-based broadcast encryption(IBBE) scheme that is CCA secure is proposed. The new scheme has many advantages over the available, such as constant size private keys and constant size ciphertexts, which solve the trade-off between the private keys size and ciphertexts size. In addition, under the standard model, the security of the new scheme is reduced to the hardness assumption-decision bilinear Diffie-Hellman exponent problem(DBDHE). This assumption is more natural than many of the hardness assumptions recently introduced to IBBE in the standard model.

Keywords: IBBE, direct CCA technique, provable security, standard model.

1 Introduction

Identity-based encryption (IBE) was introduced by Shamir[1]. It allows for a party to encrypt a message using the recipient's identity as a public key. The ability to use identities as public keys avoids the need to distribute public key certificates. So it can simplify many applications of public key encryption (PKE) and is currently an active research area. The concept of Broadcast Encryption (BE) was introduced by Fiat and Naor. In a broadcast encryption scheme a broadcaster encrypts a message for some subset S of users who are listening on a broadcast channel. Any user in S can use his private key to decrypt the broadcast. Any user outside the privileged set S should not be able to recover the message. Recently it has been widely used in digital rights management applications such as pay-TV, multicast communication, and DVD content protection.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 348–360, 2012.

Since the first scheme appeared in 1994, many BE schemes have been proposed [2-6]. In this paper, we mainly consider the construction of the identity-based broadcast encryption(IBBE). IBBE [7-11] is a generalization of IBE. One public key can be used to encrypt a message to any possible identity in IBE schemes. But in an IBBE scheme, one public key can be used to encrypt a message to any possible group of S identities. In [7, 11], the proposed scheme was based on random oracles. In addition, the size of the ciphertexts grows linearly with the number of the users. The well known construction of IBBE was proposed by Delerablée [8]. This construction achieved constant size private keys and constant size ciphertexts. However the security of her main scheme achieved only selective-identity security(a weak security) and relied on the random oracles. In [10], a new scheme with full security was proposed. But it was impractical in real-life practice since their security relied on the complex assumptions. In addition, the work in [10] had the sublinear-size ciphertexts. Moreover, the authors in [10] used a sub-algorithm at the Encrypt phase to achieve full security which increased the computations cost. In [11,12], the authors also proposed two schemes with full security. But these schemes have a same feature that achieves only CPA security. CPA-security does not guarantee any security against chosen-ciphertext attacks(CCA), where the adversary may request decryptions even after seeing the challenge ciphertext, under the natural limitation that the adversary may not request decryption of the challenge ciphertext itself. See [15-17, 19,21] for further discussion of these definitions.

In [13], the authors declared their scheme achieved fully CCA security. Unfortunately in [14], the authors proved their scheme was even not chosen plaintext secure(CPA). Hence how to construct a CCA secure IBBE is still an interesting problem. The authors in the previous scheme claimed their scheme could be transformed to the CCA scheme. However, the transformation methods in their paper were the general methods, such as hierarchical identity-based encryption, one-time signature and MAC. These folklore construction methods lead to schemes that are somewhat inefficient in the real life. The direct chosen-ciphertext technique is needed. It has been used in [15,16,17]. In this paper, we extend this technique to IBBE. The resulting scheme is more efficient than the available.Our scheme achieves constant size ciphertexts and private keys as well as scheme in [8]. However, our proposed scheme has fully CCA security. In addition, the security of our scheme is reduced to the DBDHE assumption instead of other strong assumptions. To the best of our knowledge, it is the first efficient scheme that is direct CCA secure in the standard model.

2 Preliminaries

2.1 Bilinear Maps

Let G and G_1 be two (multiplicative) cyclic groups of prime order p and g be a generator of G. A bilinear map e is a map $e : G \times G \longrightarrow G_1$ with the following properties:

(i) bilinearity: for all $u, v \in G, a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$;
(ii) non-degeneracy: $e(g, g) \neq 1$;
(iii) computability: there is an efficient algorithm to compute $e(u, v)$ for all $u, v \in G$.

2.2 Decision Bilinear Diffie-Helllman Expoent Problem (DBDHE)

The decisional bilinear Diffie-Hellman Exponent problem has been used widely to construct encryption schemes. It is given as follows. Algorithm B is given as input a random tuple $(g, y_0, y_1, \cdots, y_n, y_{n+2}, \cdots, y_{2n+2}, T)$ where $y_i = g^{\alpha^i}, y_0 = g^c$ and $\alpha, c \in Z_p^*$. Algorithm B's goal is to output 1 when $T = e(g, y_0)^{\alpha^{n+1}} = e(g, g)^{\alpha^{n+1}c}$ and 0 otherwise. Let $TU = (g, y_0, y_1, \cdots, y_n, y_{n+2}, \cdots, y_{2n+2})$. Algorithm B that outputs $b \in \{0, 1\}$ has advantage ε in solving decision *n+1*-BDHE in G if

$$|Pr(B(TU, e(g, y_0)^{\alpha^{n+1}}) = 0) - Pr(B(TU, T) = 0)| \geq \varepsilon.$$

A weak version is used in this paper[18]. It works as follows: Algorithm B is given as input a random tuple $(g, y_0, y_1, \cdots, y_n, T)$ where $y_i = g^{\alpha^i}, y_0 = g^c$ and $\alpha, c \in Z_p^*$. Algorithm B's goal is to output 1 when $T = e(g, y_0)^{\alpha^{n+1}} = e(g, g)^{\alpha^{n+1}c}$ and 0 otherwise. We also call it decision *n+1*-BDHE problem.

Definition 1. *The (t, ε)-decisional BDHE assumption holds if no* t*-time algorithm has a non-negligible advantage ε in solving the above game.*

2.3 IBBE

An identity-based broadcast encryption scheme(IBBE) with the security parameter and the maximal size m of the target set is specified as follows.

Setup. Take as input the security parameter and output a master secret key and a public key.

Extract. Take as input the master secret key and a user identity *ID*. Extract generates a user private key d_{ID}.

Encrypt. Take as input the public key and a set of included identities $S = \{ID_1, \cdots, ID_s\}$ with $s \leq m$, and output a pair (Hdr, K), where Hdr is called the header and K is a key for the symmetric encryption scheme. Compute the encryption C_M of M under the symmetric key K and broadcasts (Hdr, S, C_M).

Decrypt. Take as input a subset S, an identity ID_i and the corresponding private key, if $ID_i \in S$, the algorithm outputs K which is then used to decrypt the broadcast body C_M and recover M.

2.4 Security Model

Following [11-13], we define the security model for IBBE as follows: Both the adversary and the challenger are given as input m, the maximal size of a set of receivers.

Setup: The challenger runs Setup to obtain a public key PK and sends it to A.

Query phase 1: The adversary A adaptively issues queries $q_1, \cdots, q_{s0}$, where q_i is one of the following:

- Extraction query (ID_i): The challenger runs Extract on ID_i and sends the resulting private key to the adversary.
- Decryption query(ID_i, S, Hdr): The challenger responds with $Decrypt$ $(S, ID_i, d_{ID_i}, Hdr, PK)$.

Challenge: When A decides that phase 1 is over, A outputs two same-length messages M_0, M_1 and a challenge identity S^*. The challenger picks a random $b \in \{0, 1\}$ and sets the challenge ciphertext $C^* = Encrypt(params, M_b, S^*)$. The challenger returns C^* to A.

Note that in this paper, we consider the hybrid encryption. In the encryption phase, the encrypted message is a symmetrical key. Hence the challenge can be modified as follows. When A decides that phase 1 is over, the challenger runs Encrypt algorithm to obtain $(Hdr^*, K) = Encrypt(S^*, PK)$. The challenger then randomly selects $b \in \{0, 1\}$, sets $K_b = K$, and sets K_{1-b} to a random value. The challenger returns (Hdr^*, K_0, K_1) to A.

Query Phase 2: The adversary continues to issue queries $q_{s0+1}, \cdots, q_t$, where q_i is one of the following:

- Extraction query(ID_i), as in phase 1 with the constraint that $ID_i \notin S^*$.
- Decryption query $Hdr \neq Hdr*$ for any identity of S^*.

Guess: Finally, the adversary A outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$.

Let t denote the total number of extraction queries during the game. The advantage of A in winning the game is defined as follows [8]:

$$Adv_{IBBE}(t, m, A) = |P(b = b') - 1/2|$$

We call an adversary A in the above game a IND-ID-CCA adversary.

Definition 2. *An IBBE scheme is said to be (t, m)-IND-ID-CCA secure if $Adv_{IBBE}(t, m, A)$ is neglectable.*

2.5 Target Collision Resistant Hashing

A target collision resistant hash function (TCRHF)[19], also known as a universal one-way hash function (UOWHF), is a keyed hash function $H : K \times M_{in} \rightarrow M_{out}$ keyed by $k \in K$, where M_{in} is an input space, M_{out} is an output space and K is a hash-key space. Target collision resistance of a keyed hash function is defined using the following TCR game between the adversary and the TCR challenger:

Step 1. The adversary outputs $m_1 \in M_{in}$.

Step 2. The challenger selects random $k \in K$, and sends this to the adversary.

Step 3. The adversary outputs $m_2 \in M_{in}$. The adversary wins if $H_k(m_1) = H_k(m_2)$ and $m_2 \neq m_1$.

In this game, we call m_2 a target collision against m_1 under the key k.

Definition 3. *We say that a keyed hash function is the* (t, ϵ)*-TCRHF if no adversary running in time less than* t *can win the TCR game with probability greater than* ϵ*.*

3 New Constructions

3.1 Our Works

Setup. To generate the system parameters, PKG picks randomly $g, g_2, h_1, h_2, u_{j0}, u_{j1}, \cdots, u_{jl} \in G$ and $\alpha \in Z_p$, where $1 \leq j \leq m$. Then it sets $g_1 = g^\alpha$. The public parameters are defined as

$$PK = \{g, g_1, g_2, h_1, h_2, u_{j0}, u_{j1}, \cdots, u_{jl}, v = e(g_2, g_1)\}_{1 \leq j \leq m}$$

and the master key is g_2^α .

Extract. Given the identity $ID_i \in S$, where $S = \{ID_1, \cdots, ID_s\}$, $ID_i = (I_{i1}, \cdots, I_{il})$ and I_{ij} denotes the $\frac{n}{l} - bit$ integer in Z_p^*, PKG first computes $F_i = F(ID_i) = u_{i0} \prod_{j=1}^{l} u_{ij}^{I_{ij}}$ for $1 \leq i \leq s$. Then it selects randomly $r_i \in Z_p$ and computes the private keys as follows:

$$d_{ID_i} = (d_{i0}, d_{i1}, d_{i2} = (g_2^\alpha (F_i)^{r_i}, g^{r_i}, \prod_{j=1, j \neq i}^{s} (F_j)^{r_i}).$$

Encrypt. A broadcaster selects a random $k \in Z_p$, computes $Hdr = (C_1, C_2, C_3)$ and K as follows:

$$C_1 = g^k, C_2 = (h_1^t h_2)^k, C_3 = (\prod_{i=1}^{s} F_i)^k, K = v^k.$$

where $t = TCR(C_1)$ and TCR denotes the target collision resistant hash function.

Decrypt. In order to retrieve the message encryption key K encapsulated in the header $Hdr = (C_1, C_2, C_3)$, user with the identity ID_i and the corresponding private key $d_{ID_i} = (d_{i0}, d_{i1}, d_{i2})$ computes

$$K = \frac{e(d_{i0} d_{i2}, C_1)}{e(d_{i1}, C_3)}.$$

Correctness: If $Hdr = (C_1, C_2, C_3)$ is valid, then one can obtain

$$\frac{e(d_{i0} d_{i2}, C_1)}{e(d_{i1}, C_3)} = \frac{e(g_2^\alpha (F_i)^{r_i} \prod_{j=1, j \neq i}^{s} (F_j)^{r_i}, g^k)}{e(g^{r_i}, (\prod_{i=1}^{s} F_i)^k)} = e(g_2^\alpha, g^k) = v^k = K.$$

3.2 Efficiency

Our constructions achieve O(1)-size ciphertexts and O(1)-size private keys, which solve the trade-off of private keys and ciphertexts. In addition, v can be precomputed, so there is no pair computations at the phase of Encryption. Furthermore,

the security of the proposed scheme is reduced to the DBDHE. It is more natural than those in the existing schemes. In addition, the cost of decryption of our scheme is dominated by two pairing, which is much more efficient than that in the available. Table 1 gives the comparisons of efficiency with other schemes. From Table 1, one can find only the scheme in [13] and ours scheme have fully IND-ID-CCA security. But in [14], authors had shown the scheme in [13] was even not chosen plaintext secure(CPA).

Table 1. Comparison of Security

Schemes	Hardness assumption	security model	pk size	Ciphertext size
[8]	GBDHE	IND-sID-CPA	O(1)	O(1)
[10] 1^{st}	BDHE	IND-sID-CPA	O($\|S\|$)	O(1)
[10] 2^{nd}	BDHE	IND-sID-CPA	O(1)	O(1)
[10] 3^{rd}	BDHE	IND-ID-CPA	O(1)	Sublinear of $\|S\|$
[11]	GBDHE	IND-ID-CPA	O(1)	O(1)
[12]	Static	IND-ID-CPA	O($\|S\|$)	O(1)
[13]	TBDHE	IND-ID-**CCA**	O($\|S\|$)	O(1)
Ours	BDHE	IND-ID-**CCA**	O(1)	O(1)

3.3 Security Analysis

We give a game-based security analysis of the proposed scheme. Our proof is mainly based on the one given by Waters [20], where we make some important modifications to be able to deal with chosen-ciphertext attacks.

Theorem 1. *Under the Decisional Bilinear Diffie-Hellman exponent assumption, the IBBE is secure against chosen-ciphertext attacks.*

Proof. We will use a sequence of games to show the security of the new scheme. The first game defined will be the real identity-based encryption game. In this game, the simulator is a real oracle. Then we will change this game until the last one appears. The last one will be one in which the adversary has no advantage unconditionally. These games are run between an adversary A and a simulating algorithm B.

Game0. This is a real CCA game. In this game, we will make many conventions on how the algorithm B chooses the values appearing in the game. These conventions will be purely conceptual and, compared to the original algorithm in the previous works, do not change the distribution of any value appearing during the game. It works as follows.

Setup. B begins by choosing some values $\alpha, a, b \in Z_p$ at random. It selects randomly the elements $g, h_2, u_{j0}, u_{j1}, \cdots, u_{jl}$ and sets $g_1 = g^{\alpha}, h_1 = g^{\alpha+b}$, $g_2 = g^{\alpha^l+a}$, $v = e(g^{\alpha}, g_2)$, which implies the master key $g_2^{\alpha} = g^{a\alpha+\alpha^{l+1}}$. The public keys are

$$PK = \{g, g_1, g_2, h_1, h_2, u_{j0}, u_{j1}, \cdots, u_{jl}, v = e(g_2, g_1)\}.$$

Query Phase 1: The adversary A adaptively issues queries $q_1, \cdots, q_{s0}$, where q_i is one of the following:

- Extraction query (ID_i): The challenger runs Extract on ID_i and sends the resulting private key to the adversary.
- Decryption query(ID_i, S, Hdr): The challenger responds with $Decrypt$ $(S, ID_i, d_{ID_i}, Hdr, PK)$.

Challenge: When A decides that phase 1 is over, A outputs two same-length messages M_0, M_1 and a set of identity S^* on which it wishes to be challenged. The constraint is that the adversary does not make Extraction query for $ID_i^* \in S^*$ in Phase 1. The ciphertext is constructed as follows:

$$C_1^* = g^c, t^* = TCR(C_1^*), C_2^* = (h_1^{t^*} h_2)^c, C_3 = (\prod\nolimits_{i=1}^{s} F_i^*)^c, K^* = v^c.$$

B picks a random $b \in \{0, 1\}$, sets $K_b = K^* = v^c$ and K_{1-b} to a random value. Then B returns $(Hdr^*, K0, K1)$ to A.

Phase 2: The adversary continues to issue queries $q_{s0+1}, \cdots, q$, where qi is one of the following:

- Extraction query(ID_i), as in phase 1 with the constraint that $ID_i \notin S^*$.
- Decryption query as in phase 1 with the constraint that $S \neq S^*, Hdr \neq Hdr^*$.

Guess: Finally, the adversary A outputs a guess $b' \in \{0, 1\}$ and wins the game if $b = b'$.

For $0 \leq i \leq 8$, Let X_i denote the following event:

$$X_i : B \; wins \; the \; game_i.$$

Following the **Game0**, one can obtain $|P(X_0) - 1/2| = Adv_{IBBE}(t, q, A)$.

Game1. This game will be similar to **Game0** but hash collisions will be eliminated. It is worth noting that $C_1^* = g^c$ and $t^* = TCR(C_1^*)$ are independent of the view of A until the Guess phase runs. Hence we can assume the C_1^* and t^* have already generated at the beginning of the game.

In this game, when A issues the Decryption queries at the Query phase, B changes the responds as follows. Let $Hdr = (C_1, C_2, C_3), t = TCR(C_1)$.

If $t = t^*$ and $C_1 \neq C_1^*$, B aborts. Otherwise, it will complete this game. Let $Abort_H$ denote the aborting event. Then following [20], one can obtain

$$|\Pr(X_1) - \Pr(X_0)| \leq \Pr(Abort_H) \leq Adv_A^{TCR}(k),$$

Where $Adv_A^{TCR}(k)$ denotes the advantage of A finding the collision of TCR hash function.

Game2. We will change the game as follows. The generating method comes from the Waters's scheme[20].

Setup. Set $m = 4q$ and choose integers k_i uniformly at random between 0 and l. B then chooses random vectors $X_j = (x_{j0}, x_{j1}, \cdots, x_{jl})$ from Z_m and random vectors $Y_j = (y_{j0}, y_{j1}, \cdots, y_{jl})$ from Z_p, $1 \leq j \leq m$. Let $y_i = y^{\alpha^i}$ for $1 \leq i \leq l$. B assigns the parameters as follows.

$$u_{i0} = g^{y_{i0}} y_{l-i+1}^{p-k_i m + x_{i0}}, u_{ij} = g^{y_{ij}} y_{l-i+1}^{x_{ij}}, 1 \leq i \leq m, 1 \leq j \leq l.$$

Let

$$f_i(ID_i) = y_{i0} + \sum\nolimits_{j=1}^{l} I_{ij} y_{ij},$$

$$J_i(ID_i) = p - k_i m + x_{i0} + \sum\nolimits_{j=1}^{l} I_{ij} x_{ij}.$$

Then

$$F_i = u_{i0} \prod\nolimits_{j=1}^{l} u_{ij}^{I_{ij}} = g^{f_i(ID_i)} y_{l-i+1}^{J_i(ID_i)}.$$

The public keys are

$$PK = \{g, g_1, g_2, h_1, h_2, u_{j0}, u_{j1}, \cdots, u_{jl}, v = e(g_2, g_1)\}.$$

From the A's view, it is not different from the Game1. So one can obtain

$$\Pr(X_2) = \Pr(X_1).$$

Game3. In this game, according to the proof of [20], we will add the Forced abort. Let

$$K_i(ID_i) = \begin{cases} 0 \; if x_{i0} + \sum_{j=1}^{l} I_{ij} x_{ij} = 0 \bmod m \\ 1 \qquad\qquad otherwise \end{cases}$$

In the Query phase and Challenge phase, B will abort if $K_i(ID_i) = 0$ and $K_i(ID_i^*) \neq 0$. Following [20], we have

$$\Pr(X_3) = \Pr(X_2 \cap \overline{Abort}) + \frac{1}{2} \Pr(Abort).$$

Of course, we can add the other forced abort: Artificial Abort[13].

One can obtain $\Pr(X_2 \cap \overline{Abort}) \geq \Pr(X_2) \cdot \Pr(\overline{Abort})$. Following [17,20], we can obtain the upper bound of abortion is $\frac{1}{2(4lq2^{\frac{n}{l}})^s}$. By using the techniques in [20],

We can obtain

$$|\Pr(X_2) - (2(4lq2^{\frac{n}{l}})^s) \Pr(X_3)| \leq \frac{\rho}{2}.$$

where ρ will be given in the end of the proof.

Game4. In this game, We will changes the private key generations. Suppose A issues a private key query for an identity $ID_i \in S$. If $K_i(ID_i) = 0$, then B aborts. Otherwise, B selects a random $r'_i \in Z_p$ and sets the private keys as follows.

$$d_{ID_i} = (d_{i0}, d_{i1}, d_{i2})(g_2^{\alpha}(F_i)^{r_i}, g^{r_i}, \prod_{j=1,j\neq i}^{s}(F_j)^{r_i})$$

where $r_i = r'_i - \frac{\alpha^i}{J_i(ID_i)}$. In fact, according the Game2,

$$\begin{aligned} d_{i0} = g_2^{\alpha}(F_i)^{r_i} &= g^{a\alpha+\alpha^{l+1}}(g^{f_i(ID_i)}y_{l-i+1}^{J_i(ID_i)})^{r'_i-\frac{\alpha^i}{J_i(ID_i)}} \\ &= g^{a\alpha}y_{l+1}(g^{f_i(ID_i)}y_{l-i+1}^{J_i(ID_i)})^{r'_i}(g^{f_i(ID_i)}y_{l-i+1}^{J_i(ID_i)})^{-\frac{\alpha^i}{J_i(ID_i)}} \\ &= y_1^a y_{l+1}(g^{f_i(ID_i)}y_{l-i+1}^{J_i(ID_i)})^{r'_i}y_i^{-\frac{f_i(ID_i)}{J_i(ID_i)}}y_{l+1}^{-1} \\ &= y_1^a(g^{f_i(ID_i)}y_{l-i+1}^{J_i(ID_i)})^{r'_i}y_i^{-\frac{f_i(ID_i)}{J_i(ID_i)}} \end{aligned}$$

Hence B can compute it. For the similar technique, B can compute

$$(d_{i1}, d_{i2}) = (g^{r_i}, \prod_{j=1,j\neq i}^{s}(F_j)^{r_i})$$

since they do not depend on y_{l+1}.

From the adversary A's view, there are no changes from game3 to game4. So

$$\Pr(X_4) = \Pr(X_3).$$

Game5. In this game, we will continue to modify the game. The public key h_2 will be reconstructed. B selects a $\gamma \in Z_p$ and sets $h_2 = g^{\gamma}(h_1)^{-t^*}$, where $t^* = TCR(C_1^*)$. The public keys are given as follows,

$$PK = \{g, g_1, g_2, h_1, h_2, u_{j0}, u_{j1}, \cdots, u_{jl}, v = e(g_2, g_1)\}.$$

where

$$h_1 = g^{\alpha+b}, v = e(g_2, g_1), h_2 = g^{\gamma}(h_1)^{-t^*}, u_{i0} = g^{y_{i0}}y_{l-i+1}^{p-k_i m+x_{i0}}, u_{ij} = g^{y_{ij}}y_{l-i+1}^{x_{ij}},$$

B still knows the master key α. One can obtain easily $\Pr(X_5) = \Pr(X_4)$.

Game6. In this game, we will continue to add the forced aborts. When the adversary A issues Decryption for $(S, Hdr\ )$, B will abort if $K(ID_i) = 0$ and $t = t^*$.

If it happens in the guess phase, this game is indistinguishable from game5. Hence

$$\Pr(X_6) = \Pr(X_5).$$

If it happens in the Query phase, one can obtain easily

$$|\Pr(X_6) - \Pr(X_5)| \leq \frac{q}{p}.$$

Game7. In this game, we will change the response to the decryption queries.

- The adversary A will issue the private key query for $ID_i \in S$. If $K(ID_i) \neq 0$, B responds using the method in game4.
- Then A issues the decryption query for (S, Hdr) . If $K(ID_i) = 0, ID_i \in S$, B gives the responds as follows.

If $t = t^*$, B aborts. Otherwise, B verifies whether $e(C_1, \prod_{i=1}^{s} F_i) = e(g, C_3)$ holds or not. If the verification fails, B returns a random value as the session key. Otherwise, B gives the respond as the follows,

$$K = e(C_2/C_1^{\gamma}, g_2)^{\frac{1}{t-t^*}} / e(C_1^b, g_2).$$

From the A's view, this assigning is a real game as described in game6. In fact,

$$\begin{aligned}
\frac{e(d_{i0}d_{i2}, C_1)}{e(d_{i1}, C_3)} &= e(g_2^{\alpha}(F_i)^{r_i} \prod_{j=1, j\neq i}^{s} (F_j)^{r_i} \\
&= \frac{e(g_2^{\alpha}, C_1) e(\prod_{j=1}^{s} (F_j)^{r_i}, C_1)}{e(g^{r_i}, C_3)} \\
&= \frac{e(g_2, C_1^{\alpha}) e(\prod_{j=1}^{s} (F_j)^{r_i}, C_1)}{e(g^{r_i}, C_3)} \\
&= \frac{(e(C_2/C_1^{\gamma}, g_2)^{\frac{1}{t-t^*}} / e(C_1^b, g_2)) e(\prod_{j=1}^{s} (F_j)^{r_i}, C_1)}{e(g^{r_i}, C_3)}.
\end{aligned}$$

Note that $(C_2/C_1^{\gamma})^{\frac{1}{t-t^*}} = ((h_1^t g^{\gamma} (h_1)^{-t^*})^k)/C_1^{\gamma})^{\frac{1}{t-t^*}} = C_1^b C_1^{\alpha}$.

If $e(C_1, \prod_{i=1}^{s} F_i) = e(g, C_3)$, then

$$\frac{(e(C_2/C_1^{\gamma}, g_2)^{\frac{1}{t-t^*}} / e(C_1^b, g_2)) e(\prod_{j=1}^{s} (F_j)^{r_i}, C_1)}{e(g^{r_i}, C_3)} = K.$$

Otherwise,

$$\frac{(e(C_2/C_1^{\gamma}, g_2)^{\frac{1}{t-t^*}} / e(C_1^b, g_2)) e(\prod_{j=1}^{s} (F_j)^{r_i}, C_1)}{e(g^{r_i}, C_3)}$$

is a random value in G_1.

So

$$\Pr(X_7) = \Pr(X_6).$$

Game8. In this game, the challenge ciphertexs will be changed. Given the challenge identities set S^*. If $K_i(ID_i^*) = 0$, then Hdr^* are constructed as follows.

$$C_1^* = g^c, C_2^* = (g^c)^d, C_3^* = (g^c)^{\sum_{j=1}^{s} f_i(ID_j^*)}, K = Te(y_1, g^a)$$

If $T = e(g, g^c)^{\alpha^{l+1}}$, B selects $b \in \{0, 1\}$, sets $K_b = K = e(g, g^c)^{\alpha^{l+1}} e(y_1, g^a)$.

Otherwise B set K_{1-b} to a random value.

It is worth noting that in the game8 the simulator B can give all simulating value by using $(g, g^c, y_1, \cdots, y_l)$. It does not need to know α. This is a real simulation. So we have

$$\Pr(X_8) = \frac{1}{2},$$

$$|\Pr(X_8) - \Pr(X_7)| \leq Adv_B^{BDDH}.$$

The probabilities all different games are given as follows.

$$\begin{aligned}
Adv_{IBBE}(t, m, A) &= |P(X_0) - 1/2| \\
&\leq |\Pr(X_1) + Adv_A^{TCR} - \frac{1}{2}| \\
&= |\Pr(X_2) + Adv_A^{TCR} - \frac{1}{2}| \\
&\leq |(2(4lq2^{\frac{n}{l}})^s)(\Pr(X_3) - \frac{1}{2}) + \frac{\rho}{2} + Adv_A^{TCR}| \\
&\leq |(2(4lq2^{\frac{n}{l}})^s)(\Pr(X_5) - \frac{1}{2}) + \frac{\rho}{2} + Adv_A^{TCR}| \\
&\leq |(2(4lq2^{\frac{n}{l}})^s)(\Pr(X_6) + \frac{q}{p} - \frac{1}{2}) + \frac{\rho}{2} + Adv_A^{TCR}| \\
&= |(2(4lq2^{\frac{n}{l}})^s)(\Pr(X_7) + \frac{q}{p} - \frac{1}{2}) + \frac{\rho}{2} + Adv_A^{TCR}| \\
&= |(2(4lq2^{\frac{n}{l}})^s)(\Pr(X_7) + \frac{q}{p} - \Pr(X_8)) + \frac{\rho}{2} + Adv_A^{TCR}| \\
&\leq (2(4lq2^{\frac{n}{l}})^s)(Adv_B^{BDDH} + \frac{q}{p}) + \frac{\rho}{2} + Adv_A^{TCR}.
\end{aligned}$$

Let $\rho = Adv_{IBBE}(t, m, A)$. Then

$$Adv_{IBBE}(t, m, A) \leq 2(2(4lq2^{\frac{n}{l}})^s)(Adv_B^{BDDH} + \frac{q}{p}) + 2Adv_A^{TCR}.$$

4 Conclusions

In this paper, a direct chosen-ciphertext secure technique is introduced to construct a new IBBE scheme with the CCA security. It avoids the limitations in the previous works. The new scheme has constant size ciphtertext and private keys. It is more efficient than that in the previous works. In the standard model, we give the security proof by a sequence of games. However, our works also have some shortcomings. The public keys rely on the depth of the users set. In addition, the security of new scheme is reduced to a strong hardness assumption. It still leaves an open problem to construct an IBBE system with direct technique that is secure under a more standard assumption.

Acknowledgments. This paper was partially supported by the Nature Science Foundation of China under grant (61100231, 60970119, 61100165), the National Basic Research Program of China(973) under grant 2007CB311201, Natural Science Basic Research Plan in Shaanxi Province of China (Program No. 2012JQ8044) and the Fundamental Research Funds for the Central Universities of China.

References

1. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
2. Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
3. Boneh, D., Boyen, X.: Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
4. Boneh, D., Katz, J.: Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In: Menezes, A. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 87–103. Springer, Heidelberg (2005)
5. Boneh, D., Boyen, X.: Secure Identity Based Encryption Without Random Oracles. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 443–459. Springer, Heidelberg (2004)
6. Gentry, C.: Practical Identity-Based Encryption Without Random Oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
7. Mu, Y., Susilo, W., Lin, Y.-X., Ruan, C.: Identity-Based Authenticated Broadcast Encryption and Distributed Authenticated Encryption. In: Maher, M.J. (ed.) ASIAN 2004. LNCS, vol. 3321, pp. 169–181. Springer, Heidelberg (2004)
8. Delerablée, C.: Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (2007)
9. Du, X.J., Wang, Y., Ge, J.H., et al.: An ID-Based Broadcast Encryption Scheme for Key Distribution. IEEE Transactions on Broadcasting 51(2), 264–266 (2005)
10. Gentry, C., Waters, B.: Adaptive Security in Broadcast Encryption Systems (with Short Ciphertexts). In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 171–188. Springer, Heidelberg (2009)
11. Zhang, L.Y., Wu, Q., Hu, Y.P.: New constructions of identity-based broadcast encryption without random oracles. KSII Transactions on Internet and Information Systems 5(2), 428–439 (2011)
12. Zhang, L.Y., Hu, Y.P., Wu, Q.: Adaptively Secure Identity-based Broadcast Encryption with constant size private keys and ciphertexts from the Subgroups. Mathematical and Computer Modelling 55, 12–18 (2012)
13. Ren, Y.L., Gu, D.W.: Fully CCA2 secure identity based broadcast encryption without random oracles. Information Processing Letters 109, 527–533 (2009)
14. Wang, X.A., Weng, J., Yang, X.Y.: Cryptanalysis of an identity based broadcast encryption scheme without random oracles. Information Processing Letters 111, 461–464 (2011)

15. Kiltz, E.: Chosen-ciphertext secure identity-based encryption in the standard model with short ciphertexts. Cryptology ePrint Archive, Report 2006/122 (2006), http://eprint.iacr.org/
16. Kiltz, E.: Chosen-Ciphertext Security from Tag-Based Encryption. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 581–600. Springer, Heidelberg (2006)
17. Zhang, L.Y., Wu, Q., Hu, Y.P.: Direct Chosen-ciphertext technique for IBBE. Journal of Computational Information System 7(9), 3343–3350 (2011)
18. Chatterjee, S., Sarkar, P.: New Constructions of Constant Size Ciphertext HIBE Without Random Oracle. In: Rhee, M.S., Lee, B. (eds.) ICISC 2006. LNCS, vol. 4296, pp. 310–327. Springer, Heidelberg (2006)
19. Cramer, R., Shoup, V.: Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM Journal on Computing 33(1), 167–226 (2003)
20. Waters, B.: Efficient Identity-Based Encryption Without Random Oracles. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 114–127. Springer, Heidelberg (2005)
21. Canetti, R., Halevi, S., Katz, J.: Chosen-Ciphertext Security from Identity-Based Encryption. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 207–222. Springer, Heidelberg (2004)

A Communication Efficient Group Key Distribution Scheme for MANETs

Yang Yang[1,2]

[1] College of Mathematics and Computer Science, Fuzhou University, Fuzhou, 350002, China
[2] Key Lab of Information Security of Networks Systems, Fuzhou University, Fujian Province, China, 350002, China
yang.yang.research@gmail.com

Abstract. To secure the group communication in wireless ad hoc networks, a group key shared between participants is required. The special nature and the constraints posed by wireless ad hoc networks make the establishment of a group key a challenging task. The execution of a group key distribution scheme must be very carefully designed having in mind both the computation and communication efficiency. In this paper, we propose a communication efficient group key distribution scheme which adopts the Identity Based Broadcast Encryption (IBBE) methodology. No message exchange is required for the suggested scheme to establish a group key when the group members' identities are known to the broadcaster. A highlight of the scheme is that communication overhead remains unchanged as group size grows. In order for the group member to obtain his/her session key, only one bilinear pairing computation is required. Moreover, we show that the new scheme is proved secure in standard model under the truncated q-ABDHE assumption. Thus, the scheme can not only meet security demands but also efficient in terms of computational and communication. Our scheme is tested on a modern station and the experimental results demonstrate its suitability for large scale MANETs.

Keywords: Key distribution, Identity based, Information security, Public key cryptography, Standard model, Ad hoc network.

1 Introduction

Wireless ad hoc networks are a kind of new technique of wireless communications and expected to be used widely in many areas affecting our daily lives. The term "ad hoc" implies that this network is a network established for a special, often extemporaneous service customized to applications. Within this network, mobile nodes can communicate without the help of fixed base stations and switching centres. Mobile nodes within their coverage can directly communicate via wireless channels, and those outside their coverage can also communicate with the help of other intermediate mobile nodes. The great flexibility of MANETs also brings many great challenges. These networks are vulnerable to attackers

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 361–372, 2012.

since its characteristics of open medium and distributed cooperation resources. Compared with the mature security mechanism of point to point communication, there are more security problems of group communication to be solved in MANETs. A secure and efficient scheme is required to protect the content of group communication so that only intended members in this group can obtain the information. Secure delivery of multicast data can be achieved with the usage of a group key for data encryption. However, for the support of dynamic group membership, the group key has to be updated for each member join/leave and a mechanism distributing an updated group key to members is required. The dynamic character caused by dynamic group member changes poses a challenge on group key distribution research for wireless ad hoc network. Given the potentially enormous number of mobile users, scalability becomes another big problem. Moreover, nodes in wireless ad hoc networks are usually low power devices that run on battery power and become unusable after failure or energy depletion. As a result, it is necessary to design energy conservation group key distribution scheme in order to increase the overall network survivability. It is a problematic task to distribute group keys in a large ad hoc network.

1.1 Related Works

Zhou and Hass [1] proposed to utilize a Certification Authority (CA) service among nodes of the wireless ad hoc network: only selected nodes can serve as part of the certification authority and thus take part in admission decisions. However, contacting the distributed CA nodes in a MANET setting is difficult since such nodes might be many hops away. In [2], Kong developed an interesting threshold-RSA (TS-RSA) scheme for MANETs. Unfortunately, as pointed out in [3], TS-RSA is neither verifiable nor secure. An alternative threshold DSA (TS-DSA) scheme [4] provides verifiability and hence tolerates malicious insiders. However, TS-DSA is heavily interactive which requires $2t-1$ signers to issue certificates and thus become quite inefficient in MANET settings. Moreover, all these solutions require a pair of nodes to perform key exchange protocol to establish shared keys. Zhu et al [5] proposed a pair-wise key distribution scheme based on threshold secret sharing. However, two nodes need to communicate over several distinct paths to establish a shared key and the nodes are pre-configured with some secrets before deployment which is not realistic in a typical MANET environment. Since the foundational Diffie-Hellman (DH) protocol [6], several other protocols have been proposed for the group case. Steiner et al. [7–9] proposed a family of protocols known as Group Diffie-Hellman (GDH.1, GDH.2, GDH.3). In these protocols, the last group member serves as a controller and performs most of the computation; therefore it needs more energy compared with other group members. Due to the limitation of the nodes energy, the GDH protocol family is not suitable for the ad hoc networks. Perrig proposed a tree-based key agreement scheme [10]. After that, Kim et al. extended the work of [10] to design a Tree-Based Group Diffie-Hellman (TGDH) protocol in [11]. Compared with GDH protocols, it scales down the number of exponentiations and received messages required by the last group member to avoid excessive com-

putational and communication costs required by one node. But TGDH protocol still requires each group member to perform large modular exponentiations and transmit/receive long messages. So the TGDH protocol is also inadequate for ad hoc networks. After that, many works followed [12–14]. However, research on provably-secure group key distribution in concrete, realistic setting is fairly few.

In [15], Ng presented the first group key agreement scheme proven secure in a well-defined security model. Ny et al. incorporated the identity based cryptosystem with bilinear map and broadcast encryption scheme to construct a secure communication scheme for mobile ad hoc networks. In their scheme, the group members do not perform any message exchanges during the generation process of a group key. However, its security relies on random oracle. It has been shown that when the random oracles are instantiated with concrete hash functions, the resulting scheme may not be secure [16, 17]. Then, Zhang et al. [18] designed a new scheme which is proved secure in standard model rather than random oracles model. Unfortunately, those schemes suffer from long ciphertexts, i.e., the secret message broadcasted to the users will grow linearly with the number of receivers. The shortcoming will become serious with the increment of network scale.

1.2 Our Contribution

In this paper, we propose a novel key distribution scheme based on Identity-Based Broadcast Encryption (IBBE) approach, aiming at providing an efficient key distribution solution in MANETs. We do not require users to interactive when establishing a secret key. This scheme not only meets security demands of mobile ad hoc networks but also reduces the communication overhead. It combines the identity-based cryptosystem [19] with bilinear map to replace the contributory setup of a group key as in the previous protocols [9, 11]. Each group member is conceived as a broadcaster and can select the designated receivers by himself, then transmits the confidential message. The suggested scheme has the following merits.

Key Independence. Session keys are computationally independent from each other. Moreover, an attacker with multiple compromised session keys can not compute other session keys.

Dynamic. If a member decides to join or leave the network, a new group key can be easily established in our scheme.

Scalability. Given the potentially large number of mobile devices, the communication and computation overhead is almost unchanged due to the elaborative design of the suggested scheme.

Average Computation Load. Each group member is conceived as a broadcaster and the computation load is average for each receiver.

Efficient Communication. Message exchange is avoided. The suggested scheme only requires the broadcaster to send out an encapsulation of group key to establish the group session key. The receivers can derive the session key using their own private key without any message exchange with other group members.

Identity Based. Each group participant is assigned a distinguished identity. This avoids the authentication over digital signatures with certificate issued by a certification authority (CA).

Security. Only the intended receivers can derive the group session key. According to the security reduction theory, the proposed scheme is secure against chosen ciphertext attack and the security of the scheme is proved in standard model under the truncated decisional q-ABDHE assumption.

Note: Though an adversary can disguise himself with a legitimate identity, the adversary could not get the private key corresponding to the identity. Thus, the attacker could not derive the plaintext of secret message.

1.3 Road Map

The remainder of the paper is structured as follows. In section 2, we briefly outline the concept of bilinear map, the hardness assumption, the system model and related security notions. In section 3, we present our scheme for group key distribution. Section 4 shows the security and efficiency issues of the construction. Section 5 concludes this paper.

2 Preliminaries

2.1 Bilinear Map

Let G and G_1 be two (multiplicative) cyclic groups of prime order p and g be a generator of G. A bilinear map $\hat{e}$ is a map $\hat{e} : G \times G \rightarrow G_1$ with the following properties:

1. Bilinearity: for all $u, v \in G, a, b \in Z_p$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$;
2. Non-degeneracy: $\hat{e}(g, g) \neq 1$;
3. Computability: there is an efficient algorithm to compute $\hat{e}(u, v)$ for all $u, v \in G$.

The modified Weil pairing and the Tate pairing [19] are admissible maps of this kind. The security of our scheme described here relies on the hardness of the following assumption.

2.2 Hardness Assumption

Security of our scheme will be reduced to the hardness of the truncated q-ABDHE problem in the group in which the scheme is constructed. We briefly recall the definition of the truncated q-ABDHE problem [20]:

Definition 1: (Truncated Decisional q-Augmented Bilinear Diffie-Hellman Exponent Problem) Given a group G of prime order q, randomly choose generators g, h from G and choose $\alpha \in Z_q$ at random. Given a tuple $Tu = (g', g'_{q+2}, g, g_1, \cdots, g_q)$ and $Z \in G_1$, where we use g_i and g'_i to denote $g^{(\alpha^i)}$ and $g'^{(\alpha^i)}$, the truncated

decisional q-ABDHE problem is to decide whether Z equals to $\hat{e}(g', g_{q+1})$ or to a random element of G_1.

Definition 2. We say that truncated decisional (t, ε) q-ABDHE assumption holds in group G if there is no adversary running in time at most t can solve the truncated decisional q-ABDHE problem in G with an advantage at least ε.

2.3 System Model

Let the set $U = \{ID_1, ID_2, \cdots, ID_N\}$ be the group that contains all the ad hoc members. There exists a private key generator (PKG) that sets up system parameters, authenticates user's identity and generates private keys for each authorized user. We point out that the PKG's role is only to provide the necessary system parameters and distribute each user his private key, hence the PKG is not necessary to keep online after the completion of these procedures and is not required anymore by the users who want to setup a mobile ad hoc network.

Our scheme consists of the following four phases.

***Setup Phase*:** In this phase, PKG generates the public parameters PK and the master secrete key MK for the system.

***Extract Phase*:** The PKG will verify user's identity ID_i and generate the corresponding private key d_{ID_i} after the successful verification of ID_i.

***Encrypt Phase*:** In this phase, we consider the situation where a group of users $S = \{ID_1, \cdots, ID_n\}$ are selected to be the receivers using their wireless devices. A session key (group key) K should be established for the group. After knowing the receivers' identities, the broadcaster will generate an encapsulation header Hdr for the group key K, then broadcast (S, Hdr) in the open environment.

***Decrypt Phase*:** After receiving (S, Hdr), the intended users with identity $ID_i \in S$ could derive the group key K with his own private key d_{ID_i}. For the users with identity $ID_i \notin S$, he will get no information about K.

After the session key K is set up for the dynamic ad hoc group, the messages M can be encrypted with K to ciphertext C through efficient symmetric encryption algorithm, such as DES or AES. Moreover, the ciphertext C can only be deciphered by the users in this group.

2.4 Security Model

We assume that there exists an adversary $\mathcal{A}$. All messages available in the network are also available to $\mathcal{A}$. The main goal of $\mathcal{A}$ is to attack the scheme by decrypting any messages sent in the network intended to any set of users in but not him. $\mathcal{A}$ is considered to be successful if he wins the following interactive game.

***Setup*:** Challenger $\mathcal{C}$ runs the setup algorithm and sends adversary $\mathcal{A}$ the public parameters PK.

***Query Phase*:** Adversary $\mathcal{A}$ issues private key extract queries and decryption queries.

- *Extract queries*: $\mathcal{A}$ issues private key extract queries for any identity ID_i. In response, $\mathcal{C}$ runs *Extract* algorithm on ID_i and sends the resulting private key d_{ID_i} to adversary $\mathcal{A}$.
- *Decryption queries*: $\mathcal{A}$ issues decryption queries of (ID_i, S, Hdr) and $ID_i \in S$. Challenger $\mathcal{C}$ responds with $K = Decrypt(S, ID_i, d_{ID_i}, Hdr, PK)$.

Challenge: When $\mathcal{A}$ decides that phase 1 is over, adversary $\mathcal{A}$ outputs two equal length key K_0, K_1 and a challenge set of identities $S^* = (ID_1^*, \cdots, ID_n^*)$ with $n \leq N$. Then challenger $\mathcal{C}$ randomly selects $b \in \{0, 1\}$ and runs *Encrypt* algorithm to obtain (Hdr^*, K_b). The challenger $\mathcal{C}$ returns Hdr^* to $\mathcal{A}$.

Phase 2: Adversary $\mathcal{A}$ continues to issue private key extract queries and decryption queries as in phase 1 with the constraint that $Hdr \neq Hdr^*$ in decryption queries.

Guess: Finally, the adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins the game if $b' = b$.

If $\mathcal{A}$ somehow manages to guess the correct answer in the experiment above, then $\mathcal{A}$ wins the experiment and the scheme is not secure. We say that $\mathcal{A}$ has a guessing advantage ϵ (i.e., the probability of $\mathcal{A}$ winning the experiment) is $Pr[b = b'] = 1/2 + \epsilon$.

Definition 4: We say that a scheme is $(t, \varepsilon, q_E, q_D)$-IND-ID-CCA secure if all t time adversaries making at most q_E private key extract queries, q_D decryption queries have advantage at most ε in winning the above game.

3 Efficient Key Distribution Scheme for MANET

Inspired by Gentry's Identity Based Encryption (IBE) scheme [20], we design the efficient group key distribution scheme for mobile ad hoc networks.

3.1 Setup Phase

Given the security parameter k and the maximal size N of all the MANETs members, the private key generator (PKG) chooses a group G with order p, where $|p| \leq k$.

1) Randomly choose generators g, h of G and random $\alpha \in Z_p$, compute $g_1 = g^{\alpha}$.

2) Output the public parameter $PK = (g, g_1, h)$ while master secret key $MK = \alpha$ is kept secret by PKG.

3.2 Extract Phase

Given user's identity ID_i, PKG will verify user's identity and generates the corresponding private key after the successful verification of ID_i. PKG select random $r_{ID_i} \in Z_p$. If $ID_i = \alpha$, PKG aborts. Otherwise, calculate ID_i's private key d_{ID_i}.

$$d_{ID_i} = (d_{i,0}, d_{i,1}) = (r_{ID_i}, (hg^{-r_{ID_i}})^{1/(\alpha - ID_i)})$$

3.3 Encrypt Phase

Assume that the set of receivers is $S = \{ID_1, \cdots, ID_n\}$ with $n \leq N$, the broadcaster performs as follow.

1) Randomly select session key $K \in G_1$.
2) Randomly choose $\tau \in Z_p$ and compute $Hdr = (C_0, C_1, C_2, C_3)$ as follow

$$C_0 = K \cdot \hat{e}(g,h)^{-\tau},\ \ C_1 = g^{-\tau \cdot \prod_{j=1}^{n} ID_j},\ \ C_2 = \hat{e}(g,g)^{\tau},\ \ C_3 = g_1^{\tau}$$

3) Output (S, Hdr) and broadcast it in the system.

3.4 Decrypt Phase

Suppose that the user with identity $ID_i \in S$ has received (S, Hdr), the user compute

$$K = C_0 \cdot \hat{e}(C_3 \cdot {C_1}^{1/\prod_{j=1, j\neq i}^{n} ID_j}, d_{i,1}) C_2^{d_{i,0}}$$

After the session key K is set up for the dynamic ad hoc group, the messages M can be encrypted with K to ciphertext C which can only be deciphered by the users in the group.

Discussion: A mobile ad hoc network is dynamic and hence the group S changes whenever a mobile ad hoc network changes. When the member joins or leaves, the proposed scheme only needs to add or exclude that member's ID during execution of *Encryption Phase* in order to establish a new session key. In addition, the suggested scheme is efficient in computation, where the decryption only requires one bilinear pair computation. Further, our scheme has a constant size of ciphertext which is a desirable property for large scale network.

4 Analysis of the Proposed Scheme

4.1 Correctness

The correctness of the scheme is verified as follow:

$$\begin{aligned} & C_0 \cdot \hat{e}(C_3 \cdot {C_1}^{1/\prod_{j=1, j\neq i}^{n} ID_j}, d_{i,1}) C_2^{d_{i,0}} \\ &= C_0 \cdot \hat{e}(g_1^{\tau} g^{-\tau \cdot ID_i}, (hg^{-r_{ID_i}})^{1/(\alpha - ID_i)}) \cdot \hat{e}(g,g)^{\tau r_{ID_i}} \\ &= C_0 \cdot \hat{e}(g^{\tau}, hg^{-r_{ID_i}}) \cdot \hat{e}(g,g)^{\tau r_{ID_i}} \\ &= C_0 \cdot \hat{e}(g,h)^{\tau} = K \end{aligned}$$

4.2 Efficiency

This group key distribution scheme has been implemented in C language using the PBC (Pairing-based Cryptography) Library [21]. We choose the type-A elliptic curve parameter, which provides 1024-bit discrete log security strength equivalently and the group order is 160-bit. All experiments are run on a PC running Windows XP with Pentium Dual core CPU (3.3GHz) and a 2.0 GB of the memory.

We show in Fig. 1 and Table. 1 the execution time in setup, extraction, encryption and decryption phase under different n values (n: the size of receiver group). It can be seen that the cost of time keeps stable in each phase when n grows from 1 to 10000. Thus, the proposed scheme achieves high efficiency and is suitable for large scale mobile ad hoc network.

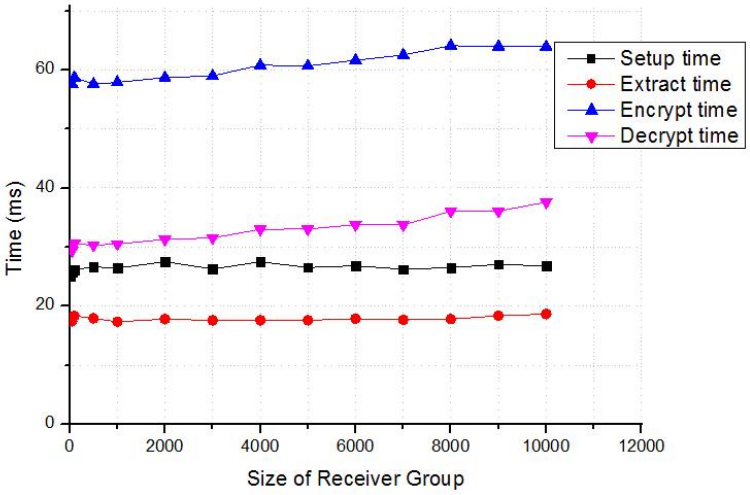


Fig. 1. Execution Time of the Scheme

4.3 Security Analysis

Theorem 1. Suppose the truncated decisional (t', ϵ) q-ABDHE assumption holds in G, then the above scheme is (t, ϵ, q_E, q_D) IND-ID-CPA secure, where $t' = t + O(\sigma \cdot q^2)$ and σ is the time for an exponential operation in G.

Proof: Suppose there exists a $(t, \varepsilon, q_E, q_D)$ adversary $\mathcal{A}$ against our scheme, then we construct an algorithm $\mathcal{C}$ that solves the truncated decisinal q-ABDHE problem with probability at least ε and in time at most t'. The challenger $\mathcal{C}$ is given a tuple of the truncated decisional q-ABDHE problem.

$$Tu = (g', g'_{q+2}, g, g_1, \cdots, g_q, Z)$$

where Z is either $\hat{e}(g_{q+1}, g')$ or a random element of G_1 and $g_i = g^{\alpha^i}$,$g'_i = (g')^{\alpha^i}$. The game between $\mathcal{A}$ and $\mathcal{C}$ proceeds as follows:

Setup:

Challenger $\mathcal{C}$ generates a random polynomial $f(x) \in Z_p[x]$ of degree q. It sets $h = g^{f(\alpha)}$, computing h from $(g, g_1, \cdots, g_q)$. It sends the public key (g, g_1, h) to $\mathcal{A}$. Since g, α and $f(x)$ are chosen uniformly at random, h is uniformly random and this public key has a distribution identical to that in the actual construction.

Table 1. Performance of the Scheme

Size of Group	Setup (ms)	Extract (ms)	Encrypt (ms)	Decrypt(ms)
10	25.015	17.540	57.898	29.175
50	25.733	17.447	57.586	29.410
100	26.148	18.327	58.716	30.549
500	26.633	17.919	57.598	30.331
1000	26.435	17.346	57.878	30.522
2000	27.522	17.799	58.694	31.317
3000	26.272	17.588	59.001	31.545
4000	27.487	17.605	60.787	32.996
5000	26.526	17.608	60.672	33.050
6000	26.814	17.863	61.638	33.788
7000	26.240	17.674	62.526	33.731
8000	26.516	17.792	64.100	36.059
9000	27.045	18.368	63.958	36.071
10000	26.812	18.649	63.909	37.609

Phase 1

• ***Extract queries***. Upon receiving a request on identity ID_i, $\mathcal{C}$ responds to a query on as follows.

1) If $ID_i = \alpha$, $\mathcal{C}$ uses α to solve truncated decision q-ABDHE immediately.

2) Else, let $F_{ID_i}(x)$ denote the $(q-1)$ degree polynomial $(f(x)-f(ID_i))/(x-ID_i)$.

3) Challenger $\mathcal{C}$ sets the private key $(d_{i,0}, d_{i,1})$ to be $(f(ID_i), g^{F_{ID_i}(\alpha)})$. This is a valid private key for ID, since

$$g^{F_{ID_i}(\alpha)} = g^{(f(\alpha)-f(ID_i))/(\alpha-ID_i)} = (hg^{-f(ID_i)})^{1/(\alpha-ID_i)}.$$

Thus, $\mathcal{C}$ has successfully simulated the private key $d_{ID_i} = (d_{i,0}, d_{i,1})$.

Challenge: Adversary $\mathcal{A}$ outputs a challenge set of identities $S^* = (ID_1^*, \cdots, ID_n^*)$ with $n \leq N$. Adversary $\mathcal{A}$ outputs two keys $K_0, K_1 \in G_1$ of equal length, $\mathcal{C}$ randomly chooses $b \in \{0, 1\}$. Let $f_2(x) = x^{q+2}$ and let

$$F_2(x) = (f_2(x) - f_2(\prod\nolimits_{i=1}^{n} ID_i^*))/x,$$

which is a polynomial of degree $q + 1$. $\mathcal{C}$ sets

$$C_3{}^* = g'^{\,f_2(\alpha)-f_2(\prod_{i=1}^{n} ID_i^*)}$$
$$C_2{}^* = Z \cdot \hat{e}(g', \prod\nolimits_{i=0}^{q} g^{F_{2,i}\cdot\alpha^i})$$
$$C_1{}^* = g'^{(-F_2(\alpha))\cdot\prod_{j=1}^{n} ID_j}$$
$$C_0{}^* = K_b/(C_2{}^*)^{f(\alpha)}$$

where $F_{2,i}$ is the coefficient of x^i in $F_2(x)$. It sends $Hdr^* = (C_0^*, C_1^*, C_2^*)$ to $\mathcal{A}$ as the challenge ciphertext. Let $\tau = (\log_g g')F_2(\alpha)$. Since $\log_g g'$ is uniformly random, τ is uniformly random.

1) If $Z = \hat{e}(g_{q+1}, g')$, then

$$\begin{aligned}
{C_3}^* &= {g'}^{F_2(\alpha)\alpha} = g^{\tau\alpha} = g_1^{\tau}, \\
{C_2}^* &= Z \cdot \hat{e}(g', \prod_{i=0}^{q} g^{F_{2,i}\cdot\alpha^i}) \\
&= \hat{e}(g', \prod_{i=0}^{q+1} g^{F_{2,i}\cdot\alpha^i}) \\
&= \hat{e}(g, g)^{\tau} \\
{C_1}^* &= {g'}^{(-F_2(\alpha))\cdot\prod_{j=1}^{n} ID_j} = g^{-\tau\cdot\prod_{j=1}^{n} ID_j}, \\
{C_0}^* &= K_b/({C_2}^*)^{f(\alpha)} = K_b/\hat{e}(g, g^{f(\alpha)})^{\tau} \\
&= K_b \cdot \hat{e}(g, h)^{-\tau}
\end{aligned}$$

Thus $Hdr^* = ({C_0}^*, {C_1}^*, {C_2}^*)$ is a valid encryption for K_b under random τ.

2) If $Z \in_R G_1$, then C_2^* is independent of b and Hdr^* is a valid encryption for K_b.

Phase 2: $\mathcal{A}$ continues to issue queries as in phase 1 with the constraint that $Hdr \neq Hdr^*$ in decryption queries.

Guess: At last, adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$. If $b' = b$, then $\mathcal{C}$ outputs 1 meaning $Z = \hat{e}(g_{q+1}, g')$. Otherwise, it outputs 0 meaning $Z \neq \hat{e}(g_{q+1}, g')$ but a random element in G_1.

1) If $Z \in_R G_1$, $\Pr[\mathcal{C}(Tu, Z) = 0] = 1/2$.

2) If $Z = \hat{e}(g_{q+1}, g')$,

$$|\Pr[\mathcal{C}(Tu, Z = \hat{e}(g_{q+1}, g')) = 0] - 1/2| \geq \varepsilon.$$

Thus,

$$\begin{aligned}
&|\Pr[\mathcal{C}(Tu, Z = \hat{e}(g_{q+1}, g')) = 0] - \Pr[\mathcal{C}(Tu, Z) = 0]| \\
&\geq \left|\left(\tfrac{1}{2} \pm \varepsilon\right) - \tfrac{1}{2}\right| = \varepsilon.
\end{aligned}$$

The time complexity of the algorithm $\mathcal{C}$ is dominated by the exponentiations performed in the preparation phase, thus $t' = t + O(\sigma \cdot q^2)$, where σ is the time of one exponentiation in G. □

5 Conclusions

In this paper, we present a communication efficient key distribution scheme for mobile ad hoc network based on identity based broadcast encryption method.

Since the group membership can change dynamically, the group key also has to be updated. An attractive property of the suggested scheme is that the proposed scheme only needs to add or exclude that member's ID during the encryption phase to generate a new group key. No matter how scalability is the group, the encapsulation of group key remains constant so that the communication overhead is unchanged. This is a highlight of our scheme that outperforms most of the previous schemes. The security analysis shows that the proposed scheme is provably secure under the truncated q-ABDHE assumption in standard model. The experiments show that the scheme achieves high efficiency and is desirable for large scale network.

Acknowledgements. This research is supported in part by National 973 program (2007CB311201), National Natural Science Foundation of China (60970119, 61100231, 61103175, 61173151) and the Technology Innovation Platform Project of Fujian Province (2009J1007).

References

1. Zhou, L., Haas, Z.J.: Securing ad hoc networks. IEEE Network 13(6), 24–30 (1999)
2. Kong, J., Zerfos, P., Luo, H., Lu, S., Zhang, L.: Providing robust and ubiquitous security support for mobile ad-hoc networks. In: Proceedings of Ninth International Conference on Network Protocols, pp. 251–260 (2001)
3. Jarecki, S., Saxena, N., Yi, J.H.: An attack on the proactive RSA signature scheme in the URSA ad hoc network access control protocol. In: Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks, pp. 1–9 (2004)
4. Narasimha, M., Tsudik, G., Yi, J.H.: On the utility of distributed cryptography in P2P and MANETs: the case of membership control. In: Proceedings of 11th IEEE International Conference on Network Protocols, pp. 336–345 (2003)
5. Zhu, S., Xu, S., Setia, S., Jajodia, S.: Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach. In: Proceedings of 11th IEEE International Conference on Network Protocols, pp. 326–335 (2003)
6. Diffie, W., Hellman, M.E.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)
7. Steiner, M., Tsudik, G., Waidner, M.: Diffie-Hellman Key Distribution Extended to Group Communication. In: Proceedings of 3rd ACM Conference on Computer and Communications Security, pp. 31–37 (1996)
8. Steiner, M., Tsudik, G., Waidner, M.: CLIQUES: A New Approach to Group Key Agreement. In: IEEE, Proceedings of 18th International Conference on Distributed Computing Systems, pp. 380–387 (1998)
9. Steiner, M., Tsudik, G., Waidner, M.: Key Agreement in Dynamic Peer Groups. IEEE Transactions on Parallel and Distributed Systems 11(8), 769–780 (2002)
10. Perrig, A.: Efficient Collaborative Key Management Protocols for Secure Autonomous Group Communication. In: International Workshop on Cryptographic Techniques and E-Commerce, pp. 192–202 (1999)
11. Kim, Y., Perrig, A., Tsudik, G.: Tree-based Group Key Agreement. ACM Transactions on Information and System Security 7(1), 60–96 (2004)

12. Biswas, G.P.: Diffie-Hellman Technique: Extended to Multiple Two-party Keys and One Multi-party Key. Information Security, IET 2(1), 12–18 (2008)
13. Brecher, T., Bresson, E., Manulis, M.: Fully Robust Tree-Diffie-Hellman Group Key Exchange. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) CANS 2009. LNCS, vol. 5888, pp. 478–497. Springer, Heidelberg (2009)
14. Tseng, Y.M., Wu, T.Y.: Analysis and Improvement on a Contributory Group Key Exchange Protocol Based on the Diffie-Hellman Technique. Informatica 21(2), 247–258 (2010)
15. Ng, C.Y., Mu, Y., Susilo, W.: An Identity-based Broadcast Encryption Scheme for Mobile Ad Hoc Networks. Journal of Telecommunications and Information Technology 1, 24–29 (2006)
16. Canetti, R., Goldreich, O., Halevi, S.: The Random Oracle Methodology. Journal of the ACM 51, 557–594 (2004)
17. Bellare, M., Boldyreva, A., Palacio, A.: An Uninstantiable Random-Oracle-Model Scheme for a Hybrid-Encryption Problem. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 171–188. Springer, Heidelberg (2004)
18. Zhang, L., Hu, Y., Mu, N.: An Identity-based Broadcast Encryption Protocol for Ad Hoc Networks. In: Proceedings of the 9th International Conference for Young Computer Scientists, pp. 1619–1623 (2008)
19. Boneh, D., Franklin, M.: Identity-based Encryption from the Weil pairing. SIAM Journal on Computing 32(3), 586–615 (2003)
20. Gentry, C.: Practical Identity-Based Encryption without random oracles. In: Proceedings of 25th Annual International Cryptology Conference, pp. 445–464 (2006)
21. Lynn B.: The PBC library, `http://crypto.stanford.edu/pbc/`

Cryptanalysis of Exhaustive Search on Attacking RSA

Mu-En Wu[1], Raylin Tso[2], and Hung-Min Sun[3]

[1] Institute of Information Science, Academia Sinica, Taipei, Taiwan
[2] Department of Computer Science, National Chengchi University, Taipei, Taiwan
[3] Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
mn@iis.sinica.edu.tw,
raylin@cs.nccu.edu.tw,
hmsun@cs.nthu.edu.tw

Abstract. In RSA equation: $ed = k \cdot \phi(N) + 1$, we may guess on partial bits of d or $p + q$ by doing an exhaustive search to further extend the security boundary of d. In this paper, we discuss the following question: Does guessing on $p + q$ bring more benefit than guessing on d? We provide the detailed analysis on this problem by using the lattice reduction technique. Our analysis shows that leaking partial most significant bits (MSBs) of $p + q$ in RSA risks more than leaking partial MSBs of d. This result inspires us to further extend the boundary of the Boneh-Durfee attack to $N^{0.284+\Delta}$, where "Δ" is contributed by the capability of exhaustive search. Assume that doing an exhaustive search for 64 bits is feasible in the current computational environment, the boundary of the Boneh-Durfee attack should be raised to $d < N^{0.328}$ for an 1024-bit RSA modulus. This is a 37 bits improvement over Boneh and Durfee's boundary.

Keywords: RSA, partial key exposure (PKE) attack, lattice basis reduction, LLL algorithm.

1 Introduction

RSA [5] is the most widely used public key cryptosystem in the world. Although this system is popular, the encryption and decryption require taking heavy exponential multiplications modulus of a large integer N, which is the product of two large primes p and q. Throughout this paper, we assume N is of 1024 bits, with p and q are of 512 bits for the reason of the security consideratioin. In general, the RSA encryption and decryption time are roughly proportional to the number of bits in public and private exponents, respectively. To reduce the encryption time (or the signature-verification time), one may wish to use a small public exponent e. The smallest possible value for e is 3, however, it has been proven to be insecure against some small public exponent attacks [2], [3]. A frequently accepted and used public exponent is $e = 2^{16} + 1 = 65537$. On the other hand, to reduce the decryption time (or the signature-generation time), one may also wish to use a short private exponent d. The key generation algorithm should be modified to choosing

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 373–379, 2012.

d first and then calculate the corresponding e satisfying $ed \equiv 1 \bmod \phi(N)$. We call this RSA variant RSA-Small-d throughout the paper. One of the drawbacks of RSA-Small-d is that the use of short private exponent encounters a serious security problem due to some powerful short private exponent attacks. In 1990, Wiener [7] first announced an attack on short private exponent, called continued fraction attack. He showed that RSA-Small-d is insecure if $d < \frac{1}{3}N^{0.25}$. Verheul and van Tilborg [6] generalized Wiener's attack to the case where one guesses high-order bits of the prime factors. Their attack needs to do an exhaustive search for about $2r + 8$ bits, where $r = \log_2(d/N^{0.25})$. If the complexity of 2^{64} (which corresponds to $r = 28$) is feasible in the current computational environment, then the security boundaryof private exponent in RSA-Small-d can be raised to $d < 2^{28}N^{0.25}$. This is a 28-bit improvement over Wiener's boundary. However, this is not the best result to the small exponent attacks in RSA-Small-d. In 1998, based on the lattice reduction technique, Boneh and Durfee [1] proposed a new attack on the use of short private exponent. They improved Wiener's bound up to $N^{0.292}$, *i.e.*, RSA-Small-d can be totally broken if $d < N^{0.292}$. This gives a 43-bit improvement over Wiener's bound if N is the size of 1024 bits

In this paper, we follow Verheul and van Tilborg's [6] idea to generalize the Boneh-Durfee attack. We try to improve Boneh and Durfee's boundary by guessing on most significant bits of d or $p + q$. We investigate the question in the following:

In RSA equation:

$$ed \equiv 1 \mod \phi(N),$$

where $\phi(N) = (N + 1) - (p + q)$ is the Euler function of N. What is the security boundary of private exponent in RSA-Small-d when ℓ MSBs of $p + q$ or d are exposed?

We take the advantage of the lattice reduction technique to analyze the above problem. According to our result, we show that guessing on MSBs of $p+q$ brings more benefit than guessing on MSBs of d.

We should point out that similar work had been published by Ernst et. al. [4]. However, the previous research are just on the case of partial d exposure. In this paper we provide the detailed analysis to the case of partial MSBs of $p + q$ exposure. Compared to the result of partial key exposure attack on d, our result shows that leaking partial MSBs of $p + q$ leads to the higher security boundary in RSA-Small-d than leaking partial MSBs of d. Furthermore, assume that doing an exhaustive search for 64 bits is feasible in the current computational environment, the boundary of the Boneh-Durfee attack should be raised to $d < N^{0.328}$ for an 1024-bit RSA modulus.

2 Preliminaries

2.1 Review of Ernst-Jochemsz-May-de Weger PKE Attack

Theorem 1. *In RSA, let $N = pq$ be an n-bit RSA modulus, where p, q are primes of bit-length $n/2$. Let e, d satisfy $ed \equiv 1 \bmod \phi(N)$ with the size of e*

is n bits and the size of d is βn bits. Given the $(\beta-\delta)n$ MSBs of d, where $0<\delta<\beta<1$, then N can be factored in polynomial time in each condition:

1. $\delta < \frac{5}{6} - \frac{1}{3}\sqrt{1+6\beta}$;
2. $\delta < \frac{3}{16}$ as $\beta \leq \frac{11}{16}$ or $\delta < \frac{1}{3} + \frac{1}{3}\beta - \frac{1}{3}\sqrt{4\beta^2+2\beta-2}$ as $\beta \geq \frac{11}{16}$.

Throughout this paper, let χ denote the number of guessing bits on MSBs of $p+q$ or d proportional to the size of modulus N. That is

$$\chi := \frac{(\#\text{ guessing bits})}{n}.$$

We may set $\chi = (\beta - \delta)$ in Ernst-Jochemsz-May-de Weger PKE attack. Thus the inequality 1 in Theorem 1 is represented as

$$\beta \leq \frac{7}{6} - \frac{\sqrt{6\chi+7}}{3} + \chi. \tag{1}$$

In practice, χ can not be set larger than 64 due to the limit of current computational capability. This is the reason that we just consider the inequality 1 but not inequality 2. Also, setting $\chi = 0$ in (1) yields the boundary $\beta \leq 0.284$, which is actually the result of the Boneh-Durfee attack, i.e.,

$$\text{RSA-Small-}d \text{ is insecure as } d < N^{0.284}.$$

This boundary can be further raised to $d < N^{0.292}$ by using the method of geometrical matrix [1], but we do not consider this technique in the paper. In fact, our improvement is much better than this boundary.

3 Cryptanalysis of RSA with Partial MSBs of $p+q$ Exposure

Assume that we do an exhaustive search on χ MSBs of $p+q$, denoted as $\overline{\varphi}$. Thus, we have $|(p+q) - \overline{\varphi}| < e^{\frac{1}{2}-\chi}$, and the approximation to $\phi(N)$ can be estimated as $N+1-\overline{\varphi}$, denoted as A. Consider the modular polynomial

$$f(x,y) = x(A-y)+1 \mod e,$$

which is the small inverse problem proposed by Boneh and Durfee [1]. The difference is that A is set to be the approximatin to $\phi(N)$ instead of just using N.

Let (x_0, y_0) be the root of $f(x,y)$. Let X and Y be the upper bound of x_0 and y_0, where $X = e^{\beta}$, and $Y = e^{\frac{1}{2}-\chi}$.

$$\det_x = e^{\frac{m(m+1)(m+2)}{3}} \cdot X^{\frac{m(m+1)(m+2)}{3}} \cdot Y^{\frac{m(m+1)(m+2)}{6}}$$
$$\det_y = e^{\frac{tm(m+1)}{2}} \cdot X^{\frac{tm(m+1)}{2}} \cdot Y^{\frac{t(m+1)(m+t+1)}{2}}$$

Plugging in $X = e^{\beta}$ and $Y = e^{\frac{1}{2}-\chi}$ we obtain

$$\begin{aligned}\det_x &= e^{\frac{m(m+1)(m+2)}{3} + \frac{\beta m(m+1)(m+2)}{3} + (\frac{1}{2}-\chi)\frac{m(m+1)(m+2)}{6}} \\ &= e^{\frac{m(m+1)(m+2)}{12}(4\beta - 2\chi + 5)}\end{aligned}$$

and

$$\begin{aligned}\det_y &= e^{\frac{tm(m+1)}{2}} \cdot e^{\frac{\beta tm(m+1)}{2}} \cdot e^{(\frac{1}{2}-\chi)\frac{t(m+1)(m+t+1)}{2}} \\ &= e^{\frac{tm(m+1)}{2}(1+\beta)+\frac{t(m+1)(m+t+1)}{4}(1-2\chi)}\end{aligned}$$

Thus, the determinant of the lattice L is

$$\det(L) = \det_x \cdot \det_y = e^D,$$

where

$$\begin{aligned}D = &\tfrac{m(m+1)(m+2)}{12}(4\beta - 2\chi + 5) \\ &+ \tfrac{tm(m+1)}{2}(1+\beta) + \tfrac{t(m+1)(m+t+1)}{4}(1-2\chi).\end{aligned}$$

In addition, the dimension of the lattice L is

$$\omega = \tfrac{(m+1)(m+2)}{2} + t(m+1)$$

and we must find the largest β to satisfy $\det(L) < e^{m\omega}$. Consequently,

$$\begin{aligned}&\tfrac{m(m+1)(m+2)}{12}(4\beta - 2\chi + 5) \\ &+ \tfrac{tm(m+1)}{2}(1+\beta) + \tfrac{t(m+1)(m+t+1)}{4}(1-2\chi) \\ &< \tfrac{m(m+1)(m+2)}{2} + tm(m+1).\end{aligned} \tag{2}$$

After rearranging (2) to quadratic equation with variable t we get

$$\begin{aligned}&\tfrac{m(m+1)(m+2)}{12}(4\beta - 2\chi - 1) \\ &+ \left[\tfrac{m(m+1)}{2}(\beta - 1) + \tfrac{(m+1)^2}{4}(1-2\chi)\right] \cdot t \\ &+ \tfrac{m+1}{4}(1-2\chi) \cdot t^2 < 0.\end{aligned} \tag{3}$$

Note that all the terms $m^i t^j$ satisfying $(i + j < 3)$ can be eliminated if we just want to find the sufficient condition of the above inequality. Hence, (3) is simplified to

$$\tfrac{4\beta-2\chi-1}{12}m^3 + \left[(\tfrac{\beta-1}{2} + \tfrac{1-2\chi}{4})m^2\right] \cdot t + \tfrac{m}{4}(1-2\chi) \cdot t^2 < 0. \tag{4}$$

The minimal value of the left-hand side is at

$$t = \frac{-\left[(\frac{\beta-1}{2}+\frac{1-2\chi}{4})m^2\right]}{\frac{m}{2}(1-2\chi)} = \frac{-\left[\frac{2\beta-1-2\chi}{2}m\right]}{1-2\chi} = -\left[\tfrac{\beta-\chi-\frac{1}{2}}{1-2\chi}m\right].$$

Hence, applying t to (4) yields

$$\begin{aligned}&\tfrac{4\beta-2\chi-1}{12}m^3 - \left[(\tfrac{\beta-1}{2} + \tfrac{1-2\chi}{4})m^2\right] \cdot \left[\tfrac{\beta-\chi-\frac{1}{2}}{1-2\chi}m\right] \\ &+ \tfrac{m}{4}(1-2\chi) \cdot \left[\tfrac{\beta-\chi-\frac{1}{2}}{1-2\chi}m\right]^2 < 0\end{aligned}$$

After dividing by m^3 and rearranging we get

$$\tfrac{4\beta-2\chi-1}{3} < \tfrac{(\beta-\chi-\frac{1}{2})^2}{1-2\chi},$$

which is simplified to

$$(4\beta - 2\chi - 1)(1 - 2\chi) < 3(\beta - \chi - \tfrac{1}{2})^2. \tag{5}$$

Note that if we apply $\chi = 0$ to (5), we get the boundary of the Boneh-Durfee attack:

$$0 < 12\beta^2 - 28\beta + 7.$$

Hence, we show that the result of the Boneh-Durfee attack is just a special case of our attacks.

4 Improving the Boneh-Durfee Attack

Rearranging (5) we get

$$\beta \leq \tfrac{7}{6} - \tfrac{\sqrt{4\chi^2 - 16\chi + 7}}{3} - \tfrac{\chi}{3}. \tag{6}$$

On the other hand, the boundary of Ernst-Jochemsz-May-de Weger PKE Attack is

$$\beta \leq \tfrac{7}{6} - \tfrac{\sqrt{6\chi + 7}}{3} + \chi. \tag{7}$$

We draw the the curves of (6) and (7) in Figure 1. The horizontal axis means the the size of short private exponent β, where $d < N^{\beta}$. The vertical axis means the ratio of guessing bits and n. In Figure 1, "Wi" and "B-D" denote the regions of the Wiener attack and the Boneh-Durfee attack, respectively. That is, RSA-Small-d is insecure under $d < N^{0.25}$ and $d < N^{0.284}$. The region "E-J-M-dW" denotes the partial key exposure attack on most significant bits of d, which was proposed by Ernst, Jochemsz, May, and de Weger [4]. Finally, "$Sec.\ III$" denotes the insecure region that partial key exposure attack on most significant bits of $p + q$ can attack successfully. As can be seen in the figure, leaking partial most significant bits of $p + q$ in RSA risks more than leaking partial MSBs of d.

The following table shows that the revised boundary of RSA-Small-d with different capability of exhaustive search. One may observe that guessing on MSBs of $p + q$ brings more benefit than guessing on MSBs of d. Assume that doing an exhaustive search for 64 bits is feasible in the current computational environment. Then RSA-Small-d with $d < N^{0.328}$ would be insecure under this attack.

# of guessing bits	$p + q$	d
8 MSBs	0.2900	0.2896
16 MSBs	0.2954	0.2945
32 MSBs	0.3062	0.3043
64 MSBs	0.3283	0.3239
80 MSBs	0.3396	0.3338
128 MSBs	0.3750	0.3637

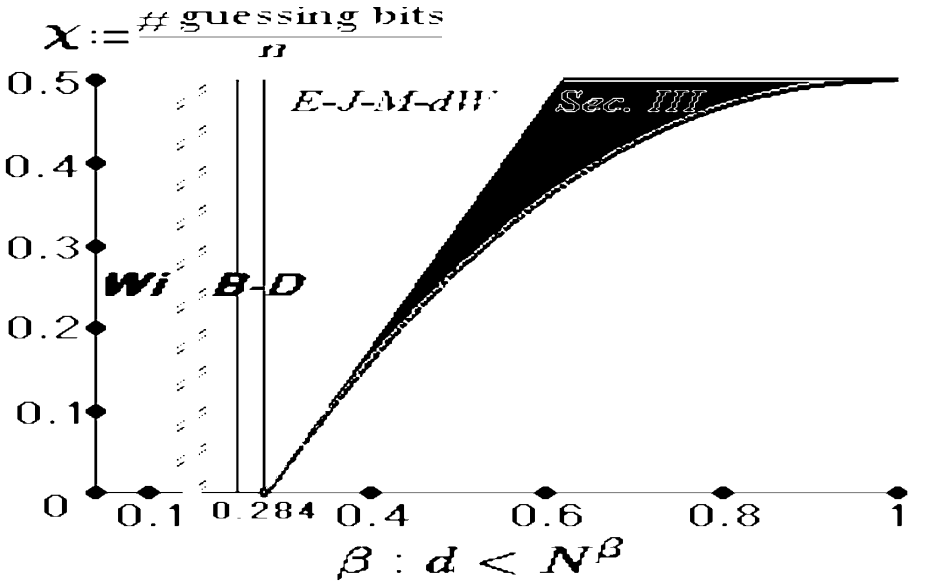


Fig. 1. The insecure region of short private exponent. The curves are two kinds of partial key exposure (PKE) attacks, which are MSBs of d and $p+q$ exposure, respectively.

5 Conclusion and Future Work

We show that in RSA system, leaking partial MSBs of $p + q$ risks more than leaking partial MSBs of d. According to our analysis, the boundary of the Boneh-Durfee attack should be raised to $d < N^{0.284+\Delta}$ by doing an exhaustive search for $n \cdot \Delta$ bits. As the development of computational technology, we may assume that doing an exhaustive search for 64 bits is feasible. Hence, for an 1024-bit RSA modulus, which is widely used now, the private exponent d shoud be chosen larger than 336 bits at least. This gives the benefit of 37 bits over the boundaey of the Boneh-Durfee attack (i.e., 299 bits, or $d < N^{0.292}$). An interesting question is how to estimate the most significant bits of $p+q$ as accurate as possible, which will bring more benefit on raising the security boundary of RSA-Small-d.

Acknowledgment. The authors would like to thank anonymous reviewers for their valuable comments and suggestions, which certainly led to improvements of this paper. This work was supported in part by the National Science Council, Taiwan, under Contract NSC 101-2628-E-004-001-MY2.

References

1. Boneh, D., Durfee, G.: Cryptanalysis of RSA with Private Key d Less than $N^{0.292}$. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 1–11. Springer, Heidelberg (1999)

2. Coppersmith, D.: Small solutions to polynomial equations, and low exponent RSA vulnerabilities. Journal of Cryptology 10, 233–260 (1997)
3. Coppersmith, D., Franklin, M., Patarin, J., Reiter, M.: Low-Exponent RSA with Related Messages. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 1–9. Springer, Heidelberg (1996)
4. Ernst, M., Jochemsz, E., May, A., de Weger, B.: Partial Key Exposure Attacks on RSA up to Full Size Exponents. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 371–386. Springer, Heidelberg (2005)
5. Rivest, R., Shamir, A., Aldeman, L.: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
6. Verheul, E., van Tilborg, H.: Cryptanalysis of less short RSA secret exponents. Applicable Algebra in Engineering, Communication and Computing 8, 425–435 (1997)
7. Wiener, M.J.: Cryptanalysis of RSA with short secret exponents. IEEE Transactions on Information Theory IT-36, 553–558 (1990)

On the Improvement of Fermat Factorization

Mu-En Wu[1], Raylin Tso[2], and Hung-Min Sun[3]

[1] Institute of Information Science, Academia Sinica, Taipei, Taiwan
[2] Department of Computer Science, National Chengchi University, Taipei, Taiwan
[3] Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
mn@iis.sinica.edu.tw, raylin@cs.nccu.edu.tw, hmsun@cs.nthu.edu.tw

Abstract. Given an integer $N = pq$, which is a product of two primes, it is difficult to determine the prime factors p and q efficiently. However, for the suitable size of a number N, Fermat's algorithm is one of the most simple method for solving it. In this paper, a method called *EPF* for estimating the prime factors of a composite number is proposed. We use the technique of continued fractions to output two integers, $p_E + q_E$ and $p_E \cdot q_E$, which are close to $p+q$ and $p \cdot q$, respectively. Furthermore, we show that *EPF* can be adopted to reduce the loop count in Fermat's algorithm before factoring a composite number. The effect depends on the size of the prime factor. We believe that there are still other applications as well wherein *EPF* can be used.

Keywords: Cryptanalysis, estimated prime factor, Fermat's algorithm, integer factorization.

1 Introduction

Lots of cryptosystems are based on the hardness of the integer factorization problem (IFP), which remains a well-studied problem [18], [22] in the past 30 years. Currently, it is suggested that the bit-length of a composite number N should be at least 1024 to be considered secure. Using the best-known factoring algorithms, the expected workload of factoring a 1024-bit number $N = pq$ is 2^{80} which is currently believed to be infeasible. In this paper we develop an approach, called "Estimated Prime Factor (*EPF*)", to estimate $p + q$, and then derive two integers p_E and q_E, which are the estimations of p and q respectively. Using *EPF*, the first 8 MSBs of $p + q$ can be efficiently determined. This result is more accurate than the traditional estimation, which estimates $p+q$ by $2\sqrt{N}$.

Furthermore, we show that *EPF* [27] can be adopted to reset the initial values of Fermat's algorithm in order to reduce the loop count. The reduced amount depends on the bit-lengths of the prime factors. Taking 16-bit p and q for experiment, the new setting reduces approximately 62% of the loop count when running Fermat's algorithm and thus enhances the efficiency by about 2.45 times.

Other than the above mentioned application, *EPF* may provide a basis for aiding cryptanalysis, such as lattice attacks [2]-[6], [9]-[12], [26], [33], integer factoring algorithms, and so on. For example, according to the cryptanalysis

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 380–391, 2012.

proposed by Weger [33], the boundary of Boneh & Durfee's lattice attack will be extended if the difference between p and q decreases. This result suggests that *EPF* afford some advantages when applied to the lattice attack.

The remainder of this paper is organized as follows: Section 2 presents the preliminaries of this paper. In Section 3, we propose the *EPF* approach for estimating the prime factors of an RSA modulus. Next, an applications of *EPF* is adopted to improve Fermat's Factorization in Section IV. Finally, we present our conclusions in Section V, where making the conclusion, along with recommendations for future work.

2 Preliminary: Fermat's Algorithm

Fermat's algorithm is a kind of methods for solving the integer factorization problem. It is well-suited for use with small computers to factor small composite numbers [20]. Without loss of generality, we suppose that the input of Fermat's algorithm is a composite integer $N = pq$ with odd numbers p and q. Note that here p and q may be composite numbers. Fermat's factoring method looks for two integers u and v such that $4N = u^2 - v^2$. Since $4N$ can be rewritten as the difference of two perfect squares, which is

$$4N = 4pq = (p+q)^2 - (p-q)^2, \tag{1}$$

we assume that $u := p + q$ and $v := p - q$ from (1). In Fermat's algorithm, setting $r = u^2 - v^2 - 4N$, we would like to find a solution (u, v) satisfying $r = u^2 - v^2 - 4N = 0$. *i.e.*,

$$4N = u^2 - v^2 = (u+v) \cdot (u-v).$$

If such (u, v) is found, then p and q can be computed by $p = \frac{u+v}{2}$ and $q = \frac{u-v}{2}$ immediately. Fermat's algorithm performs an exhaustive search to find the solution (u, v). Setting the initial values of u and v by

$$u \leftarrow 2\lceil\sqrt{N}\rceil \text{ and } v \leftarrow 0,$$

two cases making $r \neq 0$ are considered. First, if a pair $(u,\ v)$ makes $r > 0$, *i.e.*, $r = u^2 - v^2 - 4N > 0$, then it implies that v should be set larger. The next possible value of v is to set $v \leftarrow v + 2$. Note that we should not reset v by $v \leftarrow v + 1$ because $p - q$ is an even number. After resetting $v \leftarrow v + 2$, r changes to

$$r \leftarrow r - (4v + 4)\,, \tag{2}$$

where $4v + 4$ comes from the difference between the new value of v and the old one. That is,

$$(v+2)^2 - v^2 = 4v + 4.$$

Secondly, if a pair (u, v) makes $r < 0$, *i.e.*, $r = u^2 - v^2 - 4N < 0$, then it implies u should be set larger. We set $u \leftarrow u + 2$ and r changes to

$$r \leftarrow r + (4u + 4)\,, \tag{3}$$

where $4u + 4$ comes from the difference between the new value of u and the old one. That is,

$$(u + 2)^2 - u^2 = 4u + 4.$$

Repeating the processes of (2) and (3) until $r = 0$, it completes the factorization of N. We give the pseudo code of Fermat's algorithm in the following. The detail can be referenced in [8].

— **FERMAT'S ALGORITHM** —	
INITIALIZE:	READ $N = pq$
	$u \leftarrow 2\lceil\sqrt{N}\rceil$ and $v \leftarrow 0$
	$r \leftarrow u^2 - v^2 - 4N$
X_LOOP:	WHILE $r \neq 0$
	IF $r > 0$ THEN CALL Y_LOOP
	IF $r < 0$ THEN DO
	$r \leftarrow r + (4u + 4)$ and $u \leftarrow u + 2$
Y_LOOP:	WHILE $r > 0$ DO
	$r \leftarrow r - (4v + 4)$ and $v \leftarrow v + 2$
	RETURN
TERMINATE:	$p \leftarrow \frac{u+v}{2}$ and $q \leftarrow \frac{u-v}{2}$
	WRITE p, q

In Section 4, we revise the part of "INITIALIZE" in Fermat's algorithm. Through *EPF*, the new initial values u and v are adopted, instead of setting $u \leftarrow 2\lceil\sqrt{N}\rceil$ and $v \leftarrow 0$, to enhance the algorithm efficiency. We discuss the improvement later and following are the notations which will be used in the paper.

Category	Notation	
EPF	p_E:	the estimation of p.
	q_E:	the estimation of q.
	D_p:	the distance between $\sqrt{N}$ and p.
	D_q:	the distance between q and $\sqrt{N}$.
	t:	the index of continued fraction satisfying $h_t < D_p - D_q < h_{t+1}$.

3 An Approach (*EPF*) to Estimate $p + q$

In this section, a novel approach called *EPF*, which is used to estimate the prime factors of an composite number N, is proposed. *EPF* is an abbreviation for "estimated prime factor". Two cases of moduli N for *EPF* are considered:

The balanced modulus and the unbalanced modulus. Firstly, we show *EPF* when N is balanced.

3.1 Balanced Modulus

In the case of the balanced modulus $N = pq$, without loss of generality, we assume that $q < p < 2q$. Denote D_p and D_q as the distances between $\sqrt{N}$ & p, and q & $\sqrt{N}$ respectively. That is,

$$p = \sqrt{N} + D_p \text{ and } q = \sqrt{N} - D_q. \tag{4}$$

Applying (4) to $N = pq$ yields

$$N = p \cdot q = (\sqrt{N} + D_p) \cdot (\sqrt{N} - D_q) = N + \sqrt{N}(D_p - D_q) - D_p D_q. \tag{5}$$

Eliminating N in both sides of (5) yields $D_p D_q = \sqrt{N}(D_p - D_q)$, which leads to

$$\frac{1}{\sqrt{N}} = \frac{D_p - D_q}{D_p D_q}. \tag{6}$$

(6) is quite interesting because the irrational fraction $\frac{1}{\sqrt{N}}$ reveals partial information of $D_p - D_q$ and $D_p D_q$. Note that with $D_p - D_q$ and $D_p D_q$ we can compute $D_p + D_q$ by

$$(D_p + D_q)^2 = (D_p - D_q)^2 + 4D_p D_q, \tag{7}$$

and solve D_p and D_q as follows:

$$D_p = \frac{D_p + D_q}{2} + \frac{D_p - D_q}{2} \text{ and } D_q = \frac{D_p + D_q}{2} - \frac{D_p - D_q}{2}.$$

Now we use continued fractions to construct a rational sequence to approximate $\frac{1}{\sqrt{N}}$. Suppose the i^{th} convergent of the continued fraction expansion of $\frac{1}{\sqrt{N}}$ is $\frac{h_i}{k_i}$. From the property of continued fraction, we know $\frac{h_i}{k_i} \to \frac{1}{\sqrt{N}}$, as $i \to \infty$. Since the sizes of h_i and k_i grow with increase of the index i, there exists an index t such that

$$h_t < D_p - D_q < h_{t+1}. \tag{8}$$

We use h_t and k_t as the estimations of $D_p - D_q$ and $D_p D_q$ respectively instead of using the larger ones. That is,

$$h_t \approx D_p - D_q \text{ and } k_t \approx D_p D_q. \tag{9}$$

From (7), $D_p + D_q$ is estimated as

$$D_p + D_q \approx \sqrt{h_t^2 + 4k_t}. \tag{10}$$

and thus D_p and D_q are estimated as

$$D_p \approx \frac{\sqrt{h_t^2 + 4k_t} + h_t}{2} \text{ and } D_q \approx \frac{\sqrt{h_t^2 + 4k_t} - h_t}{2}.$$

Finally, we define the estimated prime factors of N as

$$p_E := \left\lceil \sqrt{N} + \frac{\sqrt{h_t^2 + 4k_t} + h_t}{2} \right\rceil \text{ and } q_E := \left\lfloor \sqrt{N} - \frac{\sqrt{h_t^2 + 4k_t} - h_t}{2} \right\rfloor.$$

3.2 Theoretical Estimation and Experimental Result on Searching the Index t

The process of computing the convergent of the continued fraction expression of $\frac{1}{\sqrt{N}}$ should stop on the index t in (8). Thus, we have to estimate the size of $D_p - D_q$ in order to determine the index t. Since $D_p < p$ and $D_q < q$, h_t should not be set larger than $\frac{n}{2}$ bits at least. Next, we investigate the method to estimate the index t theoretically and experimentally.

Theoretical Estimation: From the definitions of D_p and D_q in (4), we have

$$D_p - D_q = p + q - 2\sqrt{N} = \left(\sqrt{p} - \sqrt{q}\right)^2, \tag{11}$$

which is equivalent to

$$\log_2 (D_p - D_q) = 2 \log_2(\sqrt{p} - \sqrt{q}). \tag{12}$$

(12) shows that the bit-length of $D_p - D_q$ is twice the bit-length of $\sqrt{p} - \sqrt{q}$. Consider the following problem:

Problem: Randomly select two prime numbers p and q of $\frac{n}{2}$ bits; what is the expected value of the number that MSBs of $\sqrt{p}$ and $\sqrt{q}$ that are identical?

From our theoretical estimation (the detail is omitted due to the page limit and will be provided in the full version of the paper), the expected value is about 2.6, and it is almost independent of the bit-length of N. This implies that for any two randomly selected prime numbers p and q of $\frac{n}{2}$ bits each, the first 2.6 (expected) MSBs of $\sqrt{p}$ and $\sqrt{q}$ are identical on average. Consequently, according to (12), the size of $D_p - D_q$ is expected to be $2 \times (\frac{n}{4} - 2.6) = \frac{n}{2} - 5.2$ bits, which increases linearly with the bit-length of N.

Experimental Results: Table 1 shows the experimental results for the index t in *EPF*. Suppose that p and q are two randomly generated prime numbers of $\frac{n}{2}$ bits each; we then compute $\log_2 (D_p - D_q)$, $\log_2 (h_t)$ and $\log_2 (h_{t+1})$, which denote the bit-lengths of $D_p - D_q$, h_t and h_{t+1} respectively, where $h_{t+1} < D_p - D_q < h_t$. Each block in the table is evaluated from the average value of 1000 experimental instances. As can be observed from the first row, the bit-length of $D_p - D_q$ is approximately equal to $(\frac{n}{2} - 7)$ bits long for all n, and is greater than that of h_t by at least 1 bits on average. Note that the values of $\log_2(D_p - D_q)$ in Table 1 are slightly smaller than the theoretical estimation $\frac{n}{2} - 5.2$ bits, but the values actually increase linearly with the bit-length of N.

In *EPF*, we simply estimate the value of $D_p - D_q$, which is, however, smaller than the actual value. On the other hand, up to now, there is no theory to justify the difference between the bit-lengths of h_t and $D_p - D_q$; in fact, this would be an interesting subject of inquiry.

Table 1. The Improvement of *EPF* on $p+q$, where p and q are Balanced

n	512	1024	2048
$\log_2(D_p - D_q)$	248.476	504.626	1016.551
t (in average)	146.229	295.772	594.103
$\log_2(h_t)$	247.161	503.04	1015.201
$\log_2(h_{t+1})$	250.12	506.21	1018.14

Table 2. The Improvement of *EPF* on $p+q$, where p and q are Balanced

Balanced Modulus $N = pq$	$n = 512$	$n = 1024$	$n = 2048$
$\log_2\left((p+q) - 2\sqrt{N}\right)$	248.476	504.626	1016.551
$\log_2((p+q) - (p_E + q_E))$	247.185	503.294	1015.248

3.3 Accuracy and Further Improvement

We demonstrate the accuracy of *EPF* in Table 2. Each entry in the table is the data averaged over 1000 samples. The first row shows the difference of the bit-length between $p+q$ and its estimation by using $2\sqrt{N}$. The second row shows the difference of the bit-length between $p+q$ and its estimation by using *EPF*. As can be seen in Table 2, using $p_E + q_E$ as the estimation is more accurate than using $2\sqrt{N}$ at least one bit on average. This result shows that *EPF* is better than the traditional estimation method.

To further raise the accuracy rate of *EPF*, we may employ the properties of continued fractions. From the property of continued fraction, we know

$$h_{t+1} = a_t h_t + h_{t-1} \text{ and } k_{t+1} = a_t k_t + k_{t-1},$$

where a_t is the t^{th} component of the continued fraction expression of $\frac{1}{\sqrt{N}}$. Consequently, for any real number $\lambda \in [0,\, a_t]$, we have

$$h_t < \lambda h_t + h_{t-1} < h_{t+1} \text{ and } k_t < \lambda k_t + k_{t-1} < k_{t+1}.$$

Since $D_p - D_q$ and $D_p D_q$ are also in the intervals $(h_t,\, h_{t+1})$ and $(k_t,\, k_{t+1})$ respectively, $\lambda h_t + h_{t-1}$ and $\lambda k_t + k_{t-1}$ might be better estimations of $D_p - D_q$ and $D_p D_q$. Hence, an interesting question would be how to find a suitable value of λ that yields better estimations of $D_p - D_q$ and $D_p D_q$. Note that from the properties of continued fractions, we have

$$\text{if } t \text{ is odd, } \frac{h_{t+1}}{k_{t+1}} > \frac{1}{\sqrt{N}} > \frac{h_t}{k_t} \text{ and if } t \text{ is even, } \frac{h_{t+1}}{k_{t+1}} < \frac{1}{\sqrt{N}} < \frac{h_t}{k_t}. \tag{13}$$

(13) implies that there exists a real number $\lambda_1 \in [0,\, a_t]$ such that

$$\frac{\lambda_1 h_t + h_{t-1}}{\lambda_1 k_t + k_{t-1}} = \frac{1}{\sqrt{N}}.$$

To find an appropriate number λ, one method could be to choose λ close to λ_1, which might yield better estimations of $D_p - D_q$ and $D_p D_q$. However, we leave this concept as the subject of future work on *EPF*.

3.4 Unbalanced Modulus

In this section, we provide *EPF* such that it is suitable for an unbalanced modulus. The motivation is to look forward to applying *EPF* to afford better cryptanalysis for the second scheme, which is to be considered secure up to now, proposed by Sun *et al* [28].

Now, we assume that $N = pq$, where p and q have m bits and $(n - m)$ bits, respectively, with $m > \frac{n}{2}$. The traditional method is to estimate p and q by $N^{\frac{m}{n}}$ and $N^{1-\frac{m}{n}}$, respectively. As for *EPF* on unbalanced modulus, define $D_p := p - N^{\alpha}$ and $D_q := N^{1-\alpha} - q$, where $\alpha = \frac{m}{n}$. We have $N = pq = (N^{\alpha} + D_p) \cdot (N^{1-\alpha} - D_q)$, which is simplified to

$$N^{1-\alpha} D_p - N^{\alpha} D_q = D_p D_q. \tag{14}$$

Rearranging (14) we get

$$\frac{D_q}{D_p} + \frac{D_q}{N^{\alpha}} = \frac{N^{1-\alpha}}{N^{\alpha}}. \tag{15}$$

In order to estimate D_p and D_q in (14), we use the continued fractions to approximate $\frac{N^{1-\alpha}}{N^{\alpha}}$. Denote $\frac{B_i}{A_i}$ as the i^{th} convergent of the continued fraction expression of $\frac{N^{1-\alpha}}{N^{\alpha}}$, and t be the index satisfying $B_t < N^{1-\alpha} < B_{t+1}$.

Observing from (15), we have to construct two sequences $\{k_i\}$ and $\{h_i\}$ such that $\frac{h_i}{k_i} + \frac{h_i}{N^{\alpha}} \to \frac{N^{1-\alpha}}{N^{\alpha}}$, as $i \to \infty$.and determine the index t such that

$$\log_2 (h_t) \cong \log_2 (D_q) \text{ and } \log_2 (k_t) \cong \log_2 (D_p)$$

Now, considering the following lemma:

Lemma: Let $\frac{B_t}{A_t}$ be a rational fraction with positive integers A_t and B_t. Then the inequality $\frac{B_t - k}{A_t - k} < \frac{B_t}{A_t}$ always holds for any positive number k.

According to the above lemma, we can set $h_t = B_t - k$ and $k_t = A_t - k$ such that k is satisfying

$$\frac{B_t - k}{A_t - k} + \frac{B_t - k}{N^{\alpha}} = \frac{N^{1-\alpha}}{N^{\alpha}}. \tag{16}$$

Hence, the value of k can be solved from (16). That is,

$$k = \frac{D \pm \sqrt{D^2 - 4E}}{2}, \text{ where} \tag{17}$$

$$D = A_t + B_t + N^{\alpha} - N^{1-\alpha}, \text{ and } E = N^{\alpha} B_t - A_t N^{1-\alpha} + A_t B_t.$$

Since k should not be larger than B_t and A_t, we simply choose the smaller one of (17), *i.e.*$k = \frac{D - \sqrt{D^2 - 4E}}{2}$, as the solution of k. As a result, D_p and D_q are estimated as $A_t - k$ and $B_t - k$, respectively. And the estimated prime factors of an unbalanced modulus N are defined as $p_E := \lceil N^{\alpha} + (A_t - k) \rceil$ and $q_E := \lfloor N^{1-\alpha} - (B_t - k) \rfloor$.

4 Applying *EPF* to Fermat's Algorithm

We propose an application of *EPF* to imprve the efficiency of Fermat's algorithm by initializing the values in the begining. In Fermat's original version, $p+q$ and $p-q$ are initially estimated as $2\lceil\sqrt{N}\rceil$ and 0, respectively, and are then continuously incremented by 2 to test if they reach $p+q$ and $p-q$. Thus, the complexity of Fermat's algorithm depends on the distance between $2\lceil\sqrt{N}\rceil$ & $p+q$, and the distance between 0 & $p-q$. This suggests that if we can re-initialize Fermat's algorithm such that the u and v are much closer to $p+q$ and $p-q$ than $2\lceil\sqrt{N}\rceil$ and 0, respectively, then Fermat's algorithm will become more efficient.

Here, we employ *EPF* to initialize the values of u and v. Note that $p+q = 2\sqrt{N} + (D_p - D_q)$ and $p-q = D_p - D_q$. According to *EPF* (see (9) and (10)), $D_p - D_q \approx h_t$ and $D_p + D_q \approx \sqrt{h_t^2 + 4k_t}$, thus $p+q$ is initially estimated as $\lceil 2\sqrt{N} + h_t \rceil$ and $p-q$ is initially estimated as $\lfloor\sqrt{h_t^2 + 4k_t}\rfloor$. In addition, both in Fermat's original version and our revised version, the initial values of u and v should not be set larger than $p+q$ and $p-q$, respectively. Hence, we have to prove

$$\lceil 2\sqrt{N} + h_t \rceil < p+q \text{ and } \lfloor\sqrt{h_t^2 + 4k_t}\rfloor < p-q. \tag{18}$$

always holds.

Proof of (18): The proof of the first inequality is easy. Since $h_t < D_p - D_q$, we have

$$2\sqrt{N} + h_t < 2\sqrt{N} + D_p - D_q = p+q,$$

which completes the proof.

To prove the second inequality, we observe two relations:

$$\frac{h_t}{k_t} \approx \frac{D_p - D_q}{D_p D_q} \text{ and } h_t < D_p - D_q,$$

which imply $k_t < D_p D_q$ with high probability. Thus, we have

$$\lfloor\sqrt{h_t^2 + 4k_t}\rfloor < \lfloor\sqrt{(D_p - D_q)^2 + 4D_pD_q}\rfloor = D_p + D_q = p - q,$$

which completes the proof of the second inequality. ◇

Hence, setting the initial values

$$u \leftarrow \lceil 2\sqrt{N} + h_t \rceil \text{ and } v \leftarrow \lfloor\sqrt{h_t^2 + 4k_t}\rfloor, \tag{19}$$

in the "INITIALIZE" part of Fermat's algorithm shortens the distance between u and $\frac{p+q}{2}$, and also that between v and $\frac{p-q}{2}$. In addition, the new initial values reduce the loop count when running the algorithm and thus improve the algorithm efficiency. In what follows, we provide an example and show the experimental result.

An Example: Let $N = 1783647329$, which is the product of $p = 84449$ and $q = 21121$. First, compute each i^{th} convergent of continued fraction expression of $\frac{1}{\sqrt{N}}$, which are

$$\frac{1}{42233}, \frac{4}{168933}, \frac{273}{11529677}, \frac{277}{11698610}, \frac{827}{34926897}, \frac{1931}{81552404}, \frac{2758}{116479301}, \frac{10205}{430990307},$$

$$\frac{12963}{547469608}, \frac{49094}{2073399131}, \cdots$$

From the above sequence, it is obvious that the 9^{th} convergent, $\frac{12963}{547469608}$, is a good estimation of $\frac{D_p - D_q}{D_p D_q}$ because N is between the denominator of the 9^{th} convergent and the denominator of the 10^{th} convergent. Thus, the estimation of $D_p - D_q$, denoted as h_t, is 12963, and the estimation of $D_p D_q$, denoted as k_t, is 547469608.

Now applying h_t and k_t to Fermat's algorithm, u is set to $\lceil 2\sqrt{N} + h_t \rceil = 97430$ and v is set to $\lfloor \sqrt{h_t^2 + 4k_t} \rfloor = 48558$. We conclude the parameters in the following.

$N = 1783647329$; $p = 84449$, $q = 21121$
$h_t = 12963 < D_p - D_q \approx 21102$
$k_t = 547469608 < D_p D_q \approx 890820930$
$h_t^2 + 4k_t = 2357917801 < (D_p + D_q)^2$
$u \leftarrow \lceil 2\sqrt{N} + h_t \rceil = 97430 < p + q$
$v \leftarrow \lfloor \sqrt{h_t^2 + 4k_t} \rfloor = 48558 < p - q$

As can be seen in the above table, the initial values of u and v are smaller than $p+q$ and $p-q$, respectively. The loop count in X_LOOP is reduced from 10551 ($= \frac{(p+q)-2\lceil\sqrt{N}\rceil}{2}$) to 4070 ($= \frac{(p+q)-u}{2}$), and that in Y_LOOP is reduced from 31664 ($= \frac{p-q}{2}$) to 7385 ($= \frac{(p-q)-v}{2}$). Hence, in this case, the total loop count is reduced from 42215 ($= p - \lceil\sqrt{N}\rceil$) to 11455 ($p - \frac{(u+v)}{2}$), which is only 27.1% ($= \frac{11455}{42215}$) of the original. We show the comparisons in Table 3.

In addition, we have experimented with the case of 32-bit N, where $N = pq$ and p & q are each 16 bits long. Table 4 lists the average loop count in X_LOOP and Y_LOOP. When applying *EPF*, it is obvious that the loop count is considerably lesser than that in Fermat's algorithm. Each row in Table 4 is the data averaged over 1000 experiments. According to the experimental results, the loop count after applying *EPF* is about 38% ($= \frac{2608.45}{6870.90}$) that of the original. Furthermore, the time required is only 40.69% ($= \frac{16.61}{40.82}$) of that of the original

Table 3. Comparison of Loop Counts between Original and Revised Initial Values for $N = 1783647329$

1783647329 = 84449×21121	X_LOOP	Y_LOOP	Sum
Fermat's Algorithm	10551	31664	42215
Applying *EPF*	4070	7385	11455

Table 4. Averages in Loop Counts of Original and Revised Initial Values in Fermat's Algorithm

16-bit p and q	X_LOOP	Y_LOOP	Sum	Time (ms)
Fermat's Algorithm	622.35	6248.55	6870.90	40.82
Applying *EPF*	284.91	2323.54	2608.45	16.61

algorithm, which enhances the efficiency by 2.45 ($= \frac{40.82}{16.61}$) times. Note that Fermat's algorithm is well-suited for use in factorizing small composite numbers on devices with small computational abilities [20]. Hence, from the viewpoint of practical implementation, our improvement provides an obvious benefit under the current computational environment.

5 Discussion

In this paper, we propose a method, called *EPF*, to estimate the prime factors of a composite number $N = pq$. *EPF* can be applied to various types of brute-force attacks in order to enhance the efficiency. In particular, *EPF* can be adopted to reset the initial values of Fermat's algorithm in order to reduce the loop count and thereby enhance the algorithm efficiency when factoring a small composite number.

An interesting problem in *EPF* is whether there exists a deterministic algorithm for finding an index t satisfying $h_t < D_p - D_q < h_{t+1}$. In this paper, we use the theoretical estimation to determine the index t. The success rate is 85.1% according to our experiments. Now, another question arises — how to increase the success rate of the process of finding the index t when the deterministic algorithm is not developed. In addition, the other researchable question is how to improve the accuracy rate of MSBs of $p_E + q_E$, which brings a greater contributive effort of *EPF*.

We foresee that *EPF* can be applied to other cryptogrammic aspects, especially to the attacks for cryptosystems based on the integer factorization problem (IFP). For example, the lattice technique is commonly used for the cryptanalysis of RSA [9], [10], [12], [26], [33] or for the attacks on RSA with small exponents [2], [6], [11]. We expect *EPF* to be a supportive tool for assisting the lattice technique to increase the effort on the cryptanalysis of RSA. As the conclusion, we would like to point out that with the continuous improvements in computational capability, the security levels are expected to be higher with the assistance of *EPF*, and hence, the security analysis should be considered more carefully.

Acknowledgment. The authors would like to thank anonymous reviewers for their valuable comments and suggestions, which certainly led to improvements of this paper. This work was supported in part by the National Science Council, Taiwan, under Contract NSC 100-2628-E-007-018-MY3.

References

1. Bellare, M., Rogaway, P.: Optimal Asymmetric Encryption. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 92–111. Springer, Heidelberg (1995)
2. Bleichenbacher, D., May, A.: New Attacks on RSA with Small Secret CRT-Exponents. In: Yung, M., Dodis, Y., Kiayias, A., Malkin, T. (eds.) PKC 2006. LNCS, vol. 3958, pp. 1–13. Springer, Heidelberg (2006)
3. Blömer, J., May, A.: Low Secret Exponent RSA Revisited. In: Silverman, J.H. (ed.) CaLC 2001. LNCS, vol. 2146, pp. 4–19. Springer, Heidelberg (2001)
4. Boneh, D.: Twenty years of attacks on the RSA cryptosystem. Notices of the American Mathematical Society 46(2), 203–213 (1999)
5. Boneh, D., Durfee, G.: Cryptanalysis of RSA with private key d less than $N^{0.292}$. IEEE Trans. Inf. Theory 46(4), 1339–1349 (2000)
6. Boneh, D., Durfee, G., Frankel, Y.: An Attack on RSA Given a Small Fraction of the Private Key Bits. In: Ohta, K., Pei, D. (eds.) ASIACRYPT 1998. LNCS, vol. 1514, pp. 25–34. Springer, Heidelberg (1998)
7. Boneh, D., Shacham, H.: Fast variants of RSA. CryptoBytes 5(1), 1–9 (2002)
8. Bressoud, D.M.: Factorization and primality testing. Undergraduate Texts in Mathematics. Springer
9. Coppersmith, D.: Finding a Small Root of a Bivariate Integer Equation; Factoring with High Bits Known. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 178–189. Springer, Heidelberg (1996)
10. Coppersmith, D.: Finding a Small Root of a Univariate Modular Equation. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 155–165. Springer, Heidelberg (1996)
11. Coppersmith, D., Franklin, M., Patarin, J., Reiter, M.: Low-Exponent RSA with Related Messages. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 1–9. Springer, Heidelberg (1996)
12. Durfee, G., Nguyen, P.Q.: Cryptanalysis of the RSA Schemes with Short Private Exponent form Asiacrypt 1999. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 1–11. Springer, Heidelberg (2000)
13. Dujella, A.: Continued fractions and RSA with small private exponent. Tatra Mt. Math. Publ. 29, 101–112 (2004)
14. Galbraith, S.D., Heneghan, C., McKee, J.F.: Tunable Balancing of RSA. In: Boyd, C., González Nieto, J.M. (eds.) ACISP 2005. LNCS, vol. 3574, pp. 280–292. Springer, Heidelberg (2005)
15. Hardy, G.H., Wright, E.M.: An Introduction to the Theory of Numbers, 4th edn. Oxford Univ. Press, Cambridge (1960)
16. Hinek, M.J.: Another Look at Small RSA Exponents. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 82–98. Springer, Heidelberg (2006)
17. Galbraith, S.D., Heneghan, C., McKee, J.F.: Tunable balancing of RSA. Full version of [14]
18. Lenstra, H.W.: Factoring integers with elliptic curves. Ann. Math. 126, 649–673 (1987)
19. Lai, X.: Justified Security, speaking note. In: Proceeding of 17th Information Security Conference, Taiwan (June 2007)
20. McKee, J.: Speeding Fermat's Factoring Method. Math. Comput. 68, 1729–1738 (1999)
21. Niven, I., Zuckerman, H.S.: An Introduction to the Theory of Numbers. Wiley, Chichester (1991)

22. Pollard, J.M.: Theorems on factorization and primality testing. Proc. Cambriage Philosophical Soc. 76, 521–528 (1974)
23. Rivest, R., Shamir, A., Aldeman, L.: A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21(2), 120–126 (1978)
24. Sun, H.-M., Hinek, M.J., Wu, M.-E.: On the design of Rebalanced-RSA. Technical Report CACR 2005-35, Centre for Applied Cryptographic Research, 2005-35
25. Sun, H.-M., Yang, C.-T.: RSA with Balanced Short Exponents and Its Application to Entity Authentication. In: Vaudenay, S. (ed.) PKC 2005. LNCS, vol. 3386, pp. 199–215. Springer, Heidelberg (2005)
26. Sun, H.-M., Wu, M.-E., Ting, W.-C., Jason Hinek, M.: Dual RSA and Its Security Analysis. IEEE Trans. Inf. Theory 53(8), 2922–2933 (2007)
27. Sun, H.-M., Wu, M.-E., Chen, Y.-H.: Estimating the Prime-Factors of an RSA Modulus and an Extension of the Wiener Attack. In: Katz, J., Yung, M. (eds.) ACNS 2007. LNCS, vol. 4521, pp. 116–128. Springer, Heidelberg (2007)
28. Sun, H.-M., Yang, W.-C., Laih, C.-S.: On the Design of RSA with Short Secret Exponent. In: Lam, K.-Y., Okamoto, E., Xing, C. (eds.) ASIACRYPT 1999. LNCS, vol. 1716, pp. 150–164. Springer, Heidelberg (1999)
29. Takagi, T.: Fast RSA-type Cryptosystem Modulo $p^k q$. In: Krawczyk, H. (ed.) CRYPTO 1998. LNCS, vol. 1462, pp. 318–326. Springer, Heidelberg (1998)
30. Vanstone, S.A., Zuccherato, R.J.: Short RSA keys and their generation. J. Cryptol. 8(2), 101–114 (1995)
31. Verheul, E.R., van Tilborg, H.C.A.: Cryptanalysis of RSA secret exponents. Appl. Algebra Eng. Commun. Comput. 8(5), 425–435 (1997)
32. Wiener, M.J.: Cryptanalysis of short RSA private exponents. IEEE Trans. Inf. Theory 36(3), 553–559 (1990)
33. de Weger, B.: Cryptanalysis of RSA with small prime difference. Applicable Algebra in Engineering, Communication and Computing 13, 17–28 (2002)

Impossible Differential Cryptanalysis on Tweaked E2

Yuechuan Wei[1], Xiaoyuan Yang[1], Chao Li[2], and Weidong Du[1]

[1] Electronics Technology Department, Engineering University of Armed Police Force, Xi'an, China, 710086
[2] Science College of National University of Defense Technology, Changsha, China, 410073
wych004@163.com

Abstract. E2, a 128-bit block cipher, is an AES candidate designed and submitted by NTT corporation. It employs a Feistel structure as global structure and 2-layer Substitution-Permutation Network structure in round function. The conservative structure makes E2 immune to kinds of current cryptanalysis. Previously, there is no result of impossible differential attacks on E2 since it was once supposed to have no more than 5-round impossible differential characteristic. In this paper, the immunity of tweaked E2 (E2 without initial transformation and final transformation) against impossible differential cryptanalysis is evaluated. We present many 6-round impossible differential characteristics of tweaked E2, by using one of which, we perform a 7-round attack on tweaked E2 with 128, 192 and 256 bits key and an 8-round attack on tweaked E2 with 256 bits key. The 7-round attack requires about 2^{120} chosen plaintexts and $2^{115.5}$ 7-round encryptions; the 8-round attack needs 2^{121} chosen plaintexts and less than 2^{214} 8-round encryptions.

Keywords: Block cipher, E2, Impossible differential attack, Data complexity, Time complexity.

1 Introduction

Impossible differential cryptanalysis, proposed by Biham and Knudsen respectively, was first applied to the cipher DEAL [1] and later to Skipjack [2]. The main idea is to specify a differential of probability zero over some rounds of the cipher. Then one can derive the right key by discarding the keys which lead to the impossible differential. As one of the most powerful methods for cryptanalysis, impossible differential cryptanalysis has been applied to many block ciphers, such as AES, Camellia, MISTY1 and so on [3–9].

The key step of impossible differential cryptanalysis is to retrieve the longest impossible differential. The main technique is miss-in-the-middle, namely to find two differential characteristics with probability 1 from encryption and decryption, and connect them together when there are some inconsistencies. In Indocrypt 2003, Kim *et al.* introduced the $\mathcal{U}$-method to find impossible differential characteristics of various block ciphers [10]. However, $\mathcal{U}$-method is so general

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 392–404, 2012.

that some information is often lost during calculating the impossible differentials. Some longer impossible differentials can't be found by using the $\mathcal{U}$-method.

E2 [11] is a 128-bit block cipher with a user key length of 128, 192 or 256 bits. It was designed and submitted to AES project by NTT. The design criteria of E2 are conservative, adopting a Feistel network structure as a global structure and the 2-layer SPN (Substitution-Permutation Network) structure in its round function. All operations used in the data randomization phase are byte table lookups and byte xor's except 32-bit multiplications in IT (initial transformation) and FT (final transformation), which successfully makes E2 a fast software cipher independent of target platforms. For simplicity, we denote by E2-128/192/256 the three versions of E2 that use 128, 192 and 256 key bits, respectively.

Previous works focused on E2 with 128 bits key only. A truncated differential cryptanalysis of reduced-round variants of E2 was proposed by Matsui and Tokita in [12]. They found a 7-round byte characteristic, which led to a possible attack on an 8-round E2 without IT-Function and FT-Function. In [13], Moriai *et al*. presented a more efficient 7-round truncated differential and aimed to improve the possible attack of 8-round E2 without IT/FT functions. However, they didn't present the concrete complexity, since both the complexities of the two attacks exceeded the complexity of exhaustive search of E2 with 128 bits key, and they did not confirm whether an improved attack with complexity less than $O(2^{128})$ was possible. In [14], Aoki *et al*. studied the impossible differentials of E2 for the first time. To search the impossible differentials, the authors applied the Shrinking technique, the miss-in-the-middle technique and so on. However, no impossible differential was found for E2 without IT/FT functions with more than 5 rounds. They declared that E2 was secure against cryptanalysis with impossible differential using current known techniques. In [18], the authors declared that 6-round impossible differential characteristics of E2 existed, however, still no attack results on E2 were given.

In this paper, the 6-round impossible differentials are discussed in more detail, and the security of reduced-round E2 without IT-function and FT-function against impossible differential attack is investigated. We find a series of 6-round impossible differentials which lead to attacks on this tweaked E2. The complexity of 7-round attack on tweaked E2-128/192/256 is 2^{120} plaintexts and $2^{115.5}$ encryptions; the complexity of 8-round attack on tweaked E2 is 2^{121} plaintexts and 2^{214} encryptions. Like most cryptanalytic attacks on block ciphers, it is theoretical in the sense of the magnitude of the required data and time complexity. Unlike attacks on Camellia which can add quite a number of rounds to the known 8-round impossible differential, only one or two rounds can be added to the 6-round impossible differential of E2 because the use of 2-layer SPN structure.

The paper is organized as follows: Section 2 briefly introduces some notations and the E2 block cipher. In section 3, we describe some 6-round impossible differentials. Then attacks on tweaked E2 reduced to 7 rounds and 8 rounds are discussed in section 4 and section 5 respectively. Section 6 concludes the paper and summarizes our results.

2 Preliminaries

2.1 Notations

The following describes the notations which will be used in encryption and attack.

$L_i(R_i)$: the left(right) half output of the i^{th} round;
$X_{i,j}$: the j^{th} byte of X_i;
$\Delta L_i(\Delta R_i)$: the difference of the left(right) half output of the i^{th} round;
$K_{i,j}^{(1)}$: the j^{th} byte of subkey in first layer of the i^{th} round function;
$K_{i,j}^{(2)}$: the j^{th} byte of subkey in second layer of the i^{th} round function;
$\oplus$: xor(exclusive or);
$|$: bit string concatenation.

2.2 The E2 Block Cipher

E2 iterates round function 12 times with an initial transformation (IT-Function) at the beginning and an output transformation (FT-Function) at the end. It is a Feistel cipher with 2-layer SPN structure in its round function and the linear layer used has been proved to be optimal [15, 16]. The strategy of 2-layer SPN structure was proposed in [17]. It is based on mn-bit round functions consisting of four-layers: 1st non-linear transformation layer with n parallel m-bit s-boxes, 1st linear transformation layer, 2nd non-linear transformation layer with n parallel m-bit s-boxes, and 2nd linear transformation layer (this layer is a byte transposition). [17] shows that the round function with the 2-layer SPN structure requires one-fourth as many rounds as the 1-layer SPN structure to achieve the same differential and linear probabilities.

The decryption process of E2 is the same as the encryption process except for the order of the subkeys. Fig. 1 shows the outline of the E2 encryption process, which is defined as follows.

Let P and C be the plaintext and ciphertext respectively, L_{r-1} and R_{r-1} be the left and the right input halves of the r^{th} round, and K_r be the subkey of the r^{th} round. Then the encryption process of E2 can be written as:

$$\begin{aligned} L_0|R_0 &= \mathrm{IT}(\mathrm{P}, \mathrm{K}_{13}, \mathrm{K}_{14}), \\ R_r &= L_{r-1} \oplus F(R_{r-1}, K_r) \quad (r = 1, 2 \ldots, 12), \\ L_r &= R_{r-1}, \\ C' &= R_{12}|L_{12}, \\ C &= \mathrm{FT}(\mathrm{C}', \mathrm{K}_{16}, \mathrm{K}_{15}), \end{aligned}$$

where IT and FT are key-dependent transformations. However, we will consider E2 without IT/FT functions in this paper. In the remainder, we just use E2 to refer this tweaked version of E2 for convenience.

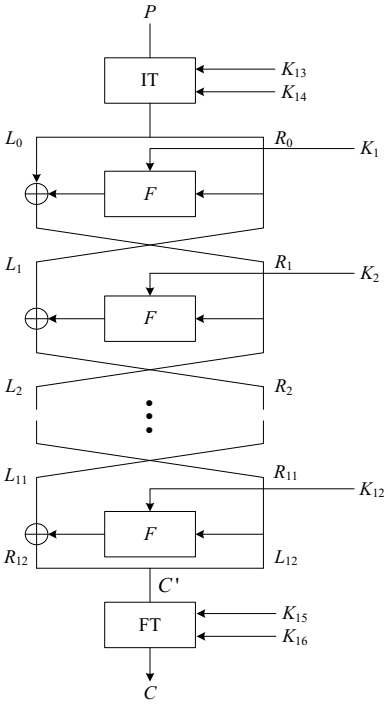


Fig. 1. Encryption Process of E2

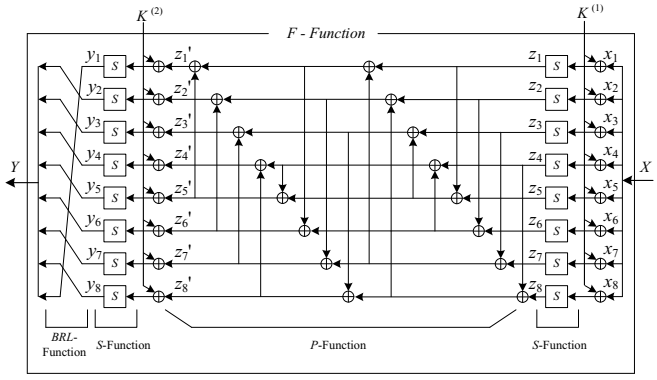


Fig. 2. Round Function of E2

Fig. 2 outlines the round function which consists of S-Function, P-Function, and BRL-Function. We refer [11] for details of the specification and notations. For readers' convenience, we give algebraic description of the variable $z_i^{'}$ in the round function in terms of the intermediate values z_i:

$$P : \mathbb{F}_{2^8}^8 \to \mathbb{F}_{2^8}^8 \colon z_1|z_2|z_3|z_4|z_5|z_6|z_7|z_8 \to z_1^{'}|z_2^{'}|z_3^{'}|z_4^{'}|z_5^{'}|z_6^{'}|z_7^{'}|z_8^{'}$$

$$\begin{aligned}
z_1^{'} &= z_2 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_6 \oplus z_7 \\
z_2^{'} &= z_1 \oplus z_3 \oplus z_4 \oplus z_6 \oplus z_7 \oplus z_8 \\
z_3^{'} &= z_1 \oplus z_2 \oplus z_4 \oplus z_5 \oplus z_7 \oplus z_8 \\
z_4^{'} &= z_1 \oplus z_2 \oplus z_3 \oplus z_5 \oplus z_6 \oplus z_8 \\
z_5^{'} &= z_1 \oplus z_2 \oplus z_4 \oplus z_5 \oplus z_6 \\
z_6^{'} &= z_1 \oplus z_2 \oplus z_3 \oplus z_6 \oplus z_7 \\
z_7^{'} &= z_2 \oplus z_3 \oplus z_4 \oplus z_7 \oplus z_8 \\
z_8^{'} &= z_1 \oplus z_3 \oplus z_4 \oplus z_5 \oplus z_8
\end{aligned}$$

The second linear layer BRL is represented as follows:

$$BRL : \mathbb{F}_{2^8}^8 \to \mathbb{F}_{2^8}^8 \colon (y_1, y_2, \ldots, y_8) \to (y_2, y_3, \ldots, y_8, y_1).$$

3 Some 6-Round Impossible Differentials

In this section, we show a series of impossible differentials of 6-round E2. An example is given in Fig. 3.

We assert that the 6-round differential

$$(0|0|0|0|a|0|0|0, 0|0|0|0|0|0|0|0) \xrightarrow{6-round} (0|0|0|0|0|0|0|0, 0|0|0|0|0|h|0|0)$$

is impossible, where a and h denote any non-zero value.

Consider an input difference $(\Delta L_0, \Delta R_0) = (0|0|0|0|a|0|0|0, 0|0|0|0|0|0|0|0)$, after passing through the first and the second round, it becomes as follows (where c_i also denotes a non-zero value):

$$\begin{aligned}
(\Delta L_1, \Delta R_1) &= (0|0|0|0|0|0|0|0, 0|0|0|0|a|0|0|0), \\
(\Delta L_2, \Delta R_2) &= (0|0|0|0|a|0|0|0, 0|c_2|c_3|c_4|0|0|c_7|c_8).
\end{aligned}$$

In the third round, after the first subkey addition and the S layer, ΔR_2 becomes $(0|c_2^*|c_3^*|c_4^*|0|0|c_7^*|c_8^*)$, where c_i^* is non-zero value. After the linear layer P it becomes $(d_1|d_2|d_3|d_4|d_5|d_6|d_7|d_8)$, thus the output difference of the second subkey addition and the S layer in the third round has the form of $(e_1|e_2|e_3|e_4|e_5|e_6|e_7|e_8)$. Whether the values of d_is and e_is $(i = 1 \ldots 8)$ are zero or not is uncertain. The BRL-function makes the output difference be $(e_2|e_3|e_4|e_5|e_6|e_7|e_8|e_1)$. Therefore the 3-round differential ends with

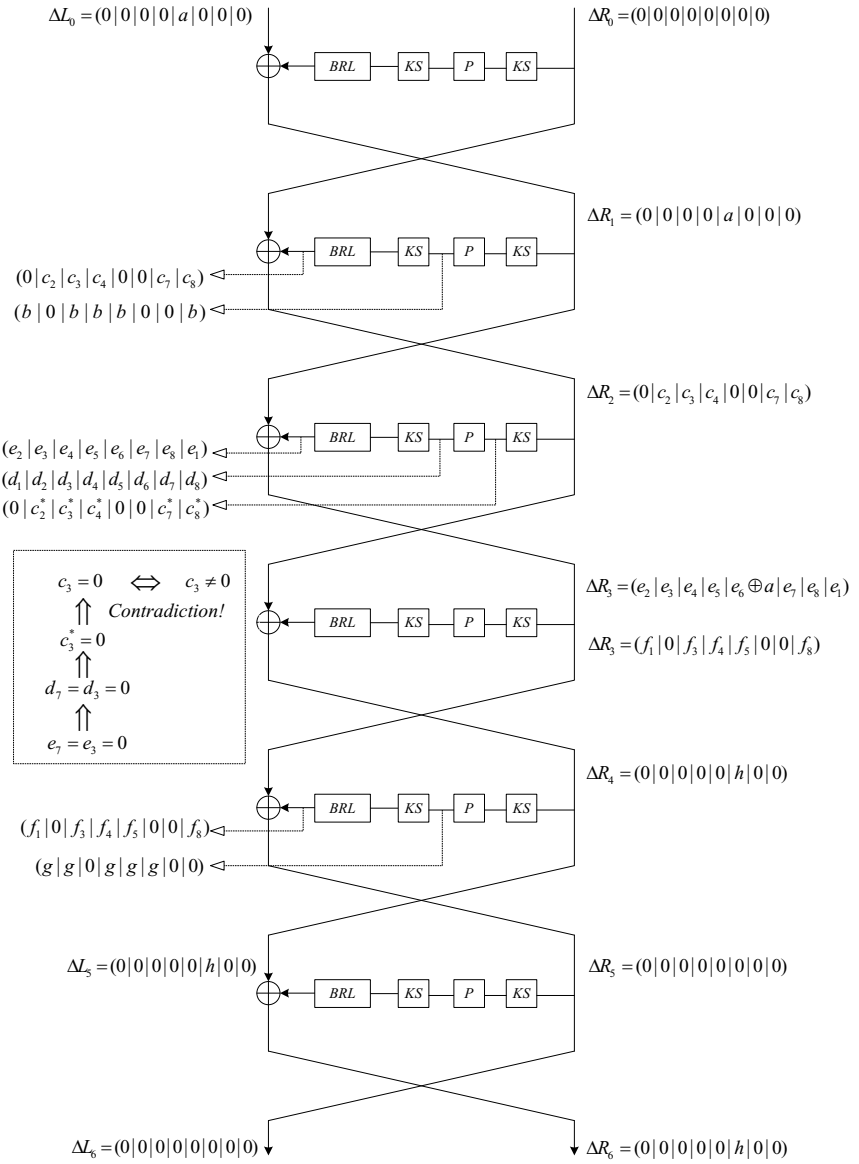


Fig. 3. 6-Round Impossible Differential of E2

$$(\Delta L_3, \Delta R_3) = (0|c_2|c_3|c_4|0|0|c_7|c_8, e_2|e_3|e_4|e_5|e_6 \oplus a|e_7|e_8|e_1).$$

Consider the other direction now. When rolling back the 6^{th} round output difference $(0|0|0|0|0|0|0|0, 0|0|0|0|0|h|0|0)$ though 3-round transformation, we get the following differences (f_i is non-zero value):

$$(\Delta L_5, \Delta R_5) = (0|0|0|0|0|h|0|0, 0|0|0|0|0|0|0|0),$$
$$(\Delta L_4, \Delta R_4) = (f_1|0|f_3|f_4|f_5|0|0|f_8, 0|0|0|0|0|h|0|0).$$

From the property of Feistel structure, we have $\Delta L_4 = \Delta R_3$, hence, $(f_1|0|f_3|f_4|f_5|0|0|f_8)$ is the same as $(e_2|e_3|e_4|e_5|e_6 \oplus a|e_7|e_8|e_1)$. So we can get $e_3 = e_7 = e_8 = 0$, thus $d_3 = d_7 = d_8 = 0$ since subkey addition and S-boxes transformations are bijective. d_i can be expressed as the linear combination of c_i^* according to the linear layer P, which implies the following equations hold (just $d_3 = d_7 = 0$ is used):

$$c_2^* \oplus c_4^* \oplus c_7^* \oplus c_8^* = 0,$$
$$c_2^* \oplus c_3^* \oplus c_4^* \oplus c_7^* \oplus c_8^* = 0.$$

From the above equations we know that c_3^* is zero, which contradicts with $c_3 \neq 0$ since subkey addition does not change the difference and S-boxes transformations are bijective.

Similarly, we can get other 6-round impossible differentials of $E2$. We define w_i as 8-byte vector, in which only the i^{th} byte is non-zero. For example, w_1 denotes $(a|0|0|0|0|0|0|0)$. If $(w_i, 0) \rightarrow (0, w_j)$ is an impossible differential, then $(w_j, 0) \rightarrow (0, w_i)$ is also an impossible differential since the encryption and the decryption are the same for Feistel cipher. The 6-round impossible differentials of E2 found by the way of Section.3 can be written as follows (for $i \leq j$).

$$\begin{array}{ll}
(w_1,0) \xrightarrow{6-round} (0,w_1), & (w_1,0) \xrightarrow{6-round} (0,w_3),\\
(w_1,0) \xrightarrow{6-round} (0,w_5), & (w_1,0) \xrightarrow{6-round} (0,w_6),\\
(w_1,0) \xrightarrow{6-round} (0,w_8), & (w_2,0) \xrightarrow{6-round} (0,w_5),\\
(w_2,0) \xrightarrow{6-round} (0,w_6), & (w_2,0) \xrightarrow{6-round} (0,w_8),\\
(w_3,0) \xrightarrow{6-round} (0,w_5), & (w_3,0) \xrightarrow{6-round} (0,w_6),\\
(w_3,0) \xrightarrow{6-round} (0,w_8), & (w_4,0) \xrightarrow{6-round} (0,w_6),\\
(w_4,0) \xrightarrow{6-round} (0,w_8), & (w_5,0) \xrightarrow{6-round} (0,w_5),\\
(w_5,0) \xrightarrow{6-round} (0,w_6), & (w_5,0) \xrightarrow{6-round} (0,w_8),\\
(w_6,0) \xrightarrow{6-round} (0,w_6), & (w_6,0) \xrightarrow{6-round} (0,w_7),\\
(w_7,0) \xrightarrow{6-round} (0,w_8). &
\end{array}$$

4 Impossible Differential Attack on E2-128/192/256 Reduced to 7 Rounds

In this section, we describe attack of E2-128/192/256 reduced to 7 rounds. The attack is based on the above 6-round impossible differential with additional

round at the beginning. A technique called "early abort" which was proposed in [5] is used in this attack. The attack is depicted in Fig. 4, and the procedure is as follows:

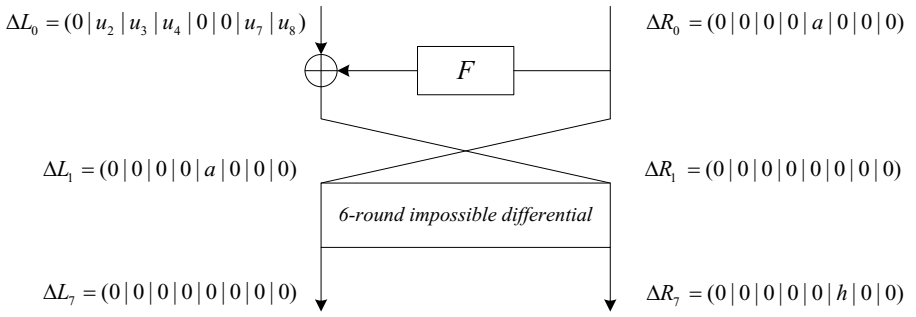


Fig. 4. 7-Round Impossible Differential Attack on Tweaked E2

Step 1. Choose structure of plaintexts as follows:

$$L_0 = (\alpha_1|x_2|x_3|x_4|\alpha_5|\alpha_6|x_7|x_8),$$
$$R_0 = (\beta_1|\beta_2|\beta_3|\beta_4|y_5|\beta_6|\beta_7|\beta_8),$$

where x_i $(i = 2, 3, 4, 7, 8)$, y_5 take all possible values in $\mathbb{F}_2^8$, α_i $(i = 1, 5, 6)$ and β_i $(i = 1, 2, 3, 4, 6, 7, 8)$ are constants in $\mathbb{F}_2^8$. For each possible value of $(x_2, x_3, x_4, x_7, x_8, y_5)$, we can get a unique 128-bit string. Hence, a structure includes 2^{48} plaintexts and there are $2^{48\times 2}/2 = 2^{95}$ plaintext pairs in a structure. So the 2^{72} structures yield a total of 2^{167} plaintext pairs.

Step 2. Keep only the pairs whose difference of ciphertexts $(\Delta L_7, \Delta R_7)$ satisfy the following:

$$\Delta L_7 = (0|0|0|0|0|0|0|0),$$
$$\Delta R_7 = (0|0|0|0|0|h|0|0),$$

where h is a unknown non-zero value. The expected number of remaining pairs is about $2^{167} \times 2^{-120} = 2^{47}$.

Step 3. Guess the 64-bit subkeys $K_1^{(1)}$ and 5 subkey bytes $K_{1,1}^{(2)}$, $K_{1,3}^{(2)}$, $K_{1,4}^{(2)}$, $K_{1,5}^{(2)}$, $K_{1,8}^{(2)}$.

Step 3.1. For every remaining pair (L_0, R_0) and (L_0^*, R_0^*), guess the 64-bit subkey $K_1^{(1)}$ and compute

$$Z_1 = PS(R_0 \oplus K_1^{(1)}),$$
$$Z_1^* = PS(R_0^* \oplus K_1^{(1)}).$$

Step 3.2. Guess the 5 bytes of $K_1^{(2)}$,

(a) Guess the 8-bit value for the $K_{1,1}^{(2)}$, and for each guess compute:

$$q_1 = s(Z_{1,1} \oplus K_{1,1}^{(2)}) \oplus s(Z_{1,1}^* \oplus K_{1,1}^{(2)}) \oplus L_{0,8} \oplus L_{0,8}^*.$$

Choose pairs whose corresponding q_1 is zero. The expected number of the remaining pairs is $2^{47} \times 2^{-8} = 2^{39}$, since the probability of such condition is 2^{-8}.

(b) Guess the 8-bit value for the $K_{1,3}^{(2)}$, and for each guess compute:

$$q_2 = s(Z_{1,3} \oplus K_{1,3}^{(2)}) \oplus s(Z_{1,3}^* \oplus K_{1,3}^{(2)}) \oplus L_{0,2} \oplus L_{0,2}^*.$$

Then check whether $q_2 = 0$ and keep only the qualified pairs. The expected number of the remaining pairs is $2^{39} \times 2^{-8} = 2^{31}$.

(c) Guess the 8-bit value for the $K_{1,4}^{(2)}$, and for each guess compute:

$$q_3 = s(Z_{1,4} \oplus K_{1,4}^{(2)}) \oplus s(Z_{1,4}^* \oplus K_{1,4}^{(2)}) \oplus L_{0,3} \oplus L_{0,3}^*.$$

Choose pairs whose corresponding q_3 is zero and the expected number of remaining pairs is $2^{31} \times 2^{-8} = 2^{23}$.

(d) Guess the 8-bit value for the $K_{1,5}^{(2)}$, and for each guess compute:

$$q_4 = s(Z_{1,5} \oplus K_{1,5}^{(2)}) \oplus s(Z_{1,5}^* \oplus K_{1,5}^{(2)}) \oplus L_{0,4} \oplus L_{0,4}^*.$$

Choose pairs whose corresponding q_3 is zero and the expected number of remaining pairs is $2^{23} \times 2^{-8} = 2^{15}$.

(e) Guess the 8-bit value for the $K_{1,8}^{(2)}$, and for each guess compute:

$$q_5 = s(Z_{1,8} \oplus K_{1,8}^{(2)}) \oplus s(Z_{1,8}^* \oplus K_{1,8}^{(2)}) \oplus L_{0,7} \oplus L_{0,7}^*.$$

Then check whether $q_5 = 0$. If yes, discard the candidate value of $(K_1^{(1)}, K_{1,i}^{(2)})$ $(i = 1, 3, 4, 5, 8)$.

Since such a difference is impossible, every key that proposes such a difference is a wrong key. After analyzing 2^{15} ciphertexts pairs, there remains only about $2^{104}(1 - 2^{-8})^{2^{15}}$ wrong candidate values of $(K_1^{(1)}, K_{1,i}^{(2)})$ $(i = 1, 3, 4, 5, 8)$, which is much less than 1.

The time complexity of Step 3.1 is about $2^{47} \times 2^{64} \times 2 = 2^{112}$ one round operations. Step 3.2(a) has a time complexity of about $2^{47} \times 2^{64} \times 2^8 \times 2 = 2^{120}$ s-box computations, which is equivalent to $2^{120} \times \frac{1}{16} = 2^{116}$ one round operations. Step 3.2(b) to (e) still need 2^{116} one round operations, respectively. The total time complexity of the attack is dominated by Step 3.2 and is about $2^{116} \times 5 = 2^{118.3}$ one round operations, which is equivalent to $2^{118.3} \times \frac{1}{7} = 2^{115.5}$ 7-round encryptions.

Therefore, this attack requires about 2^{120} chosen plaintexts and less than $2^{115.5}$ encryptions of 7-round E2. Since the time complexity is less than 2^{128}, this attack is valid for E2 with 128-bit, 192-bit and 256-bit keys.

5 Impossible Differential Attack on E2-256 Reduced to 8 Rounds

An 8-round impossible differential attack on E2-256 without IT/FT function can be obtained in this Section. In order to reduce the time complexity, we utilize some pre-computations. The attack is based on the above 6-round impossible differentials with additional one round at the beginning and one round at the end as shown in Fig. 5.

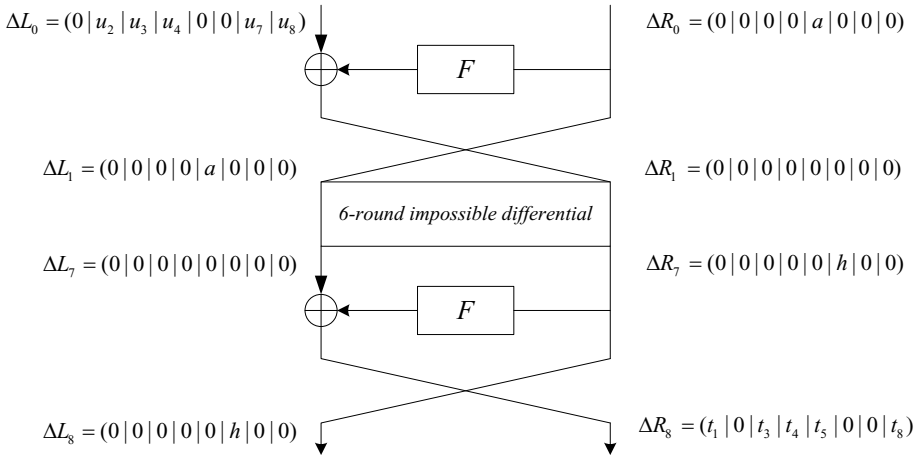


Fig. 5. 8-Round Impossible Differential Attack on Tweaked E2

The attack procedure is as follows:

Step 1. Precalculation: for s-box, define $T(\alpha, \beta) = \{x \in \mathbb{F}_2^8 | s(x \oplus \alpha) \oplus s(x) = \beta\}$, then take all possible values of (α, β), and store $T(\alpha, \beta)$ in a table.

Step 2. Choose structure of plaintexts as follows:

$$L_0 = (\alpha_1 | x_2 | x_3 | x_4 | \alpha_5 | \alpha_6 | x_7 | x_8),$$
$$R_0 = (\beta_1 | \beta_2 | \beta_3 | \beta_4 | y_5 | \beta_6 | \beta_7 | \beta_8),$$

where x_i $(i = 2, 3, 4, 7, 8)$, y_5 take all possible values in $\mathbb{F}_2^8$, α_i $(i = 1, 5, 6)$ and β_i $(i = 1, 2, 3, 4, 6, 7, 8)$ are constants in $\mathbb{F}_2^8$. For each possible value of $(x_2, x_3, x_4, x_7, x_8, y_5)$, we can get a unique 128-bit string. Hence, a structure includes 2^{48} plaintexts and there are $2^{48 \times 2}/2 = 2^{95}$ plaintext pairs in a structure. So the 2^{73} structures yield a total of 2^{168} plaintext pairs.

Step 3. Keep only the pairs whose ciphertexts differential $(\Delta L_8, \Delta R_8)$ satisfy the following:

$$\Delta L_8 = (0|0|0|0|0|h|0|0),$$
$$\Delta R_8 = (t_1|0|t_3|t_4|t_5|0|0|t_8),$$

where t_i $(i = 1, 3, 4, 5, 8)$ are unknown non-zero values. The expected number of remaining pairs is about $2^{168} \times 2^{-80} = 2^{88}$.

Step 4. Guess the 64-bit subkey $K_8^{(1)}$ and 5 subkey bytes $K_{8,1}^{(2)}$, $K_{8,2}^{(2)}$, $K_{8,4}^{(2)}$, $K_{8,5}^{(2)}$, $K_{8,6}^{(2)}$.

Step 4.1. For every remaining pair (L_0, R_0) and (L_0^*, R_0^*), guess the 64-bit subkey $K_8^{(1)}$ and compute

$$Z_8 = PS(L_8 \oplus K_8^{(1)}),$$
$$Z_8^* = PS(L_8^* \oplus K_8^{(1)}).$$

Step 4.2. Guess the 5 bytes of $K_8^{(2)}$ and compute

$$q_1 = s(Z_{8,1} \oplus K_{8,1}^{(2)}) \oplus s(Z_{8,1}^* \oplus K_{8,1}^{(2)}) \oplus R_{8,8} \oplus R_{8,8}^*,$$
$$q_2 = s(Z_{8,2} \oplus K_{8,2}^{(2)}) \oplus s(Z_{8,2}^* \oplus K_{8,2}^{(2)}) \oplus R_{8,1} \oplus R_{8,1}^*,$$
$$q_3 = s(Z_{8,4} \oplus K_{8,4}^{(2)}) \oplus s(Z_{8,4}^* \oplus K_{8,4}^{(2)}) \oplus R_{8,3} \oplus R_{8,3}^*,$$
$$q_4 = s(Z_{8,5} \oplus K_{8,5}^{(2)}) \oplus s(Z_{8,5}^* \oplus K_{8,5}^{(2)}) \oplus R_{8,4} \oplus R_{8,4}^*,$$
$$q_5 = s(Z_{8,6} \oplus K_{8,6}^{(2)}) \oplus s(Z_{8,6}^* \oplus K_{8,6}^{(2)}) \oplus R_{8,5} \oplus R_{8,5}^*.$$

Then check whether $q_i = 0$ $(1 \le i \le 5)$ and keep only the qualified pairs. Since the probability is about 2^{-40}, the expected number of the remaining pairs is $2^{88} \times 2^{-40} = 2^{48}$.

Step 5. Guess the 64-bit subkeys $K_1^{(1)}$ and 5 subkey bytes $K_{1,1}^{(2)}$, $K_{1,3}^{(2)}$, $K_{1,4}^{(2)}$, $K_{1,5}^{(2)}$, $K_{1,8}^{(2)}$, for every remaining plaintext pair (L_0, R_0) and (L_0^*, R_0^*),

$$L_0 = (\alpha_1|x_2|x_3|x_4|\alpha_5|\alpha_6|x_7|x_8),$$
$$R_0 = (\beta_1|\beta_2|\beta_3|\beta_4|y_5|\beta_6|\beta_7|\beta_8),$$
$$L_0^* = (\alpha_1|x_2^*|x_3^*|x_4^*|\alpha_5|\alpha_6|x_7^*|x_8^*),$$
$$R_0^* = (\beta_1|\beta_2|\beta_3|\beta_4|y_5^*|\beta_6|\beta_7|\beta_8),$$

perform the following:

Step 5.1. Guess $K_1^{(1)}$ and compute:

$$Z_1 = PS(R_0 \oplus K_1^{(1)}),$$
$$Z_1^* = PS(R_0^* \oplus K_1^{(1)}).$$

Step 5.2. Guess the 5 bytes of $K_1^{(2)}$ and compute

$$q_1 = s(Z_{1,1} \oplus K_{1,1}^{(2)}) \oplus s(Z_{1,1}^* \oplus K_{1,1}^{(2)}) \oplus L_{0,8} \oplus L_{0,8}^*,$$
$$q_2 = s(Z_{1,3} \oplus K_{1,3}^{(2)}) \oplus s(Z_{1,3}^* \oplus K_{1,3}^{(2)}) \oplus L_{0,2} \oplus L_{0,2}^*,$$
$$q_3 = s(Z_{1,4} \oplus K_{1,4}^{(2)}) \oplus s(Z_{1,4}^* \oplus K_{1,4}^{(2)}) \oplus L_{0,3} \oplus L_{0,3}^*,$$
$$q_4 = s(Z_{1,5} \oplus K_{1,5}^{(2)}) \oplus s(Z_{1,5}^* \oplus K_{1,5}^{(2)}) \oplus L_{0,4} \oplus L_{0,4}^*,$$
$$q_5 = s(Z_{1,8} \oplus K_{1,8}^{(2)}) \oplus s(Z_{1,8}^* \oplus K_{1,8}^{(2)}) \oplus L_{0,7} \oplus L_{0,7}^*.$$

Then check whether $q_i = 0 (1 \leq i \leq 5)$. If yes, discard the candidate value of $(K_1^{(1)}, K_{1,i}^{(2)}, K_8^{(1)}, K_{8,j}^{(2)})$ $(i = 1, 3, 4, 5, 8; j = 1, 2, 4, 5, 6)$.

Since such a difference is impossible, every key that proposes such a difference is a wrong key. After analyzing 2^{48} ciphertexts pairs, there remains only about $2^{208}(1 - 2^{-40})^{2^{48}}$ wrong candidate values of $(K_1^{(1)}, K_{1,i}^{(2)}, K_8^{(1)}, K_{8,j}^{(2)})$ $(i = 1, 3, 4, 5, 8; j = 1, 2, 4, 5, 6)$, which is much less than 1.

The time complexity of Step 4.1 requires about $2^{88} \times 2^{64} \times 2 = 2^{153}$ one round operations. The precalculation can decrease the complexity of Step 4.2, one can look up the table $T(Z_{8,k} \oplus Z_{8,k}^*, R_{8,i} \oplus R_{8,i}^*)$ to judge whether the $q_i s$ are zero or not. This step needs about $2^{88} \times 2^{64} \times 2^{40} \times 5 \approx 2^{194.3}$ table lookups. Step 5 has a time complexity of about $2^{48} \times 2^{104} \times 2^{64} \times 2 = 2^{217}$ one round operations and $2^{48} \times 2^{64} \times 2^{40} \times 5 \approx 2^{154.3}$ table lookups respectively. However, in contrast to the workload of encryptions in Step 5, the workload of table lookups in both Step 4.2 and Step 5 is negligible.

Consequently, this attack requires about 2^{121} chosen plaintexts and less than 2^{214} encryptions of 8-round E2.

6 Conclusion

The block cipher E2 was proposed as an AES candidate. It employs a Feistel structure and a 2-layer SPN structure in round function. In this paper we describe some 6-round impossible differentials of E2, and present 7-round attack and 8-round attack on E2 without IT/FT. The 7-round attack is valid for tweaked E2 with 128, 192, and 256 bits key and the 8-round is valid for tweaked E2 with 256 bits key. Cryptanalysis given in this paper is the first security evaluation of E2 against impossible differential cryptanalysis.

Acknowledgements. The authors would like to thank the anonymous referees for their valuable remarks. This work is supported by the Natural Science Foundation of China for Young Scholars (No: 61202492).

References

1. Knudsen, L.: DEAL — A 128-bit Block Cipher. Technical Report 151, Department of Informatics, University of Bergen, Bergen, Norway (1998)
2. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999)
3. Wu, W., Zhang, W., Feng, D.: Impossible Differential Cryptanalysis of Reduced-Round ARIA and Camellia. Journal of Computer Science and Technology 22(3), 449–456 (2007)
4. Wu, W., Zhang, L., Zhang, W.: Improved Impossible Differential Cryptanalysis of Reduced-Round Camellia. In: Avanzi, R., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 442–456. Springer, Heidelberg (2009)

5. Lu, J., Kim, J., Keller, N., Dunkelman, O.: Improving the Efficiency of Impossible Differential Cryptanalysis of Reduced Camellia and MISTY1. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 370–386. Springer, Heidelberg (2008)
6. Lu, J., Dunkelman, O., Keller, N., Kim, J.-S.: New Impossible Differential Attacks on AES. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 279–293. Springer, Heidelberg (2008)
7. Dunkelman, O., Keller, N.: An Improved Impossible Differential Attack on MISTY1. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 441–454. Springer, Heidelberg (2008)
8. Mala, H., Shakiba, M., Dakhilalian, M., Bagherikaram, G.: New Results on Impossible Differential Cryptanalysis of Reduced–Round Camellia–128. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 281–294. Springer, Heidelberg (2009)
9. Zhang, W., Han, J.: Impossible Differential Analysis of Reduced Round CLEFIA. In: Yung, M., Liu, P., Lin, D. (eds.) Inscrypt 2008. LNCS, vol. 5487, pp. 181–191. Springer, Heidelberg (2009)
10. Kim, J., Hong, S., Sung, J., Lee, S., Lim, J., Sung, S.: Impossible Differential Cryptanalysis for Block Cipher Structures. In: Johansson, T., Maitra, S. (eds.) INDOCRYPT 2003. LNCS, vol. 2904, pp. 82–96. Springer, Heidelberg (2003)
11. Kanda, M., Moriai, S., Aoki, K., Ueda, H., Takashima, Y., Ohta, K., Matsumoto, T.: E2–A New 128-Bit Block Cipher. IEICE Transactions Fundamentals of Electronics, Communications and Computer Sciences E83-A(1), 48–59 (2000)
12. Matsui, M., Tokita, T.: Cryptanalysis of a Reduced Version of the Block Cipher $E2$. In: Knudsen, L. (ed.) FSE 1999. LNCS, vol. 1636, pp. 71–80. Springer, Heidelberg (1999)
13. Moriai, S., Sugita, M., Aoki, K., Kanda, M.: Security of E2 against Truncated Differential Cryptanalysis. In: Heys, H., Adams, C. (eds.) SAC 1999. LNCS, vol. 1758, pp. 106–117. Springer, Heidelberg (2000)
14. Aoki, K., Kanda, M.: Search for Impossible Differential of E2, `http://csrc.nist.gov/encryption/aes/round1/comment`
15. Sugita, M., Kobara, K., Imai, H.: Pseudorandomness and Maximum Average of Differential Probability of Block Ciphers with SPN-Structures like E2. In: Proceedings of the Second Advanced Encryption Standard Candidate Conference, pp. 200–214 (1999)
16. Sugita, M.: Security of Block Ciphers with SPN-Structures. Technical Report of IEICE. ISEC 98–30
17. Kanda, M., Takashima, Y., Matsumoto, T., Aoki, K., Ohta, K.: A Strategy for Constructing Fast Round Functions with Practical Security Against Differential and Linear Cryptanalysis. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 264–279. Springer, Heidelberg (1999)
18. Wei, Y., Li, P., Sun, B., Li, C.: Impossible Differential Cryptanalysis on Feistel Ciphers with *SP* and *SPS* Round Functions. In: Zhou, J., Yung, M. (eds.) ACNS 2010. LNCS, vol. 6123, pp. 105–122. Springer, Heidelberg (2010)

Linear Cryptanalysis and Security Tradeoff of Block Ciphering Systems with Channel Errors

Jing Guo[1] and Zhuxiao Wang[2]

[1] National Computer Network Emergency Response Technical Team/Coordination Center of China (CNCERT/CC), Beijing 100029, China
guojing.research@gmail.com
[2] School of Control and Computer Engineering, State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources, North China Electric Power University, Beijing 102206, China
wangzx@ncepu.edu.cn

Abstract. Channel errors are usually treated as an obstacle in designing an encrypted wireless system. So we are supposed to reduce them as much as possible due to the potential error bursts contributed by an avalanche effect of block ciphers. In this paper, we propose that channel errors are to be explored for the benefit of security enhancement as they could be translated to additional efforts in cryptanalysis for an adversary node. To achieve this, a system with an outer block channel coder and an inner block cipher is presented. A framework for linear cryptanalysis is established under which an eavesdropper takes advantage of linear relationship among coded symbols, as well as linear approximation of ciphers. Also presented is an analysis on the tradeoff between security enhancement and performance degradation in the presence of channel errors.

Keywords: linear cryptanalysis, block cipher, channel error.

1 Introduction

One of the primary requirements in designing a block cipher is to let it have a strong avalanche effect [1]. Therefore, the traditional wisdom for encryption in wireless systems is to either put a powerful channel coding after a block cipher so that the input to the decryption block at a legitimate receiver can be considered nearly error free, like in WiMAX [2], or perform channel coding first before a stream cipher based encryption block [3,4]. The adoption of stream cipher after channel coding in GSM systems is because of its no-error-propagation property in its deciphering operation [5,6].

Both of these two approaches are essentially based on one common philosophy, namely, minimizing channel error effect from the perspective of a legitimate receiver. When channel errors cannot be eliminated completely, the detrimental consequence of error propagation on the legitimate receiver's performance after deciphering has been investigated [7,8]. Such analysis is invoked by traditional wisdom in that channel encoding and encryption operate independently.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 405–416, 2012.

Inspired by recent developments on information theoretic secrecy where channel uncertainty could be leveraged to achieve positive secrecy [9,10], we propose that channel errors should not always be considered as a pure obstacle we have to conquer. Instead, the channel errors inherent in wireless systems could be capitalized as an opportunity for us to explore to enhance security further at a cost of channel coding complexity at legitimate users. The underlying intuition is that intercepting ciphertexts distorted by channel errors also exacerbates its efforts on cryptanalysis for an eavesdropper. To the best of our knowledge, our work is the first to explicitly consider in a systematic way the relationship between linear block channel coding and block ciphering with an aim at exploring channel errors.

The paper is organized as follows. System model and problem formulation are presented in Section 2. Cryptanalysis by Eve and its associated metrics are provided in Section 3. In Section 4, we give the tradeoff analysis between security enhancement for Eve and degraded performance for Bob. Finally conclusions are drawn in Section 5.

2 System Model and Problem Formulation

In this section, we first introduce an encrypted communication system which will be analyzed to investigate the tradeoff between secrecy enhancement and performance degradation. To give an in-depth theoretical analysis, an ENM(Encrypted Nonlinear Mapping) is used instead of a real cipher in the model. An ENM is a basic module which is widely used in modern cryptography, such as DES, AES and so on. Without loss of generality, the linear cryptanalysis method suggested in this paper is also applicable to real ciphers which consist of ENMs. Then, we define "ciphertext amount" and "time complexity" as secrecy metrics, and "codeword correction probability" to evaluate the degradation of performance.

2.1 System Architecture

To investigate the secrecy enhancement introduced by channel errors, we add an encrypted nonlinear mapping after channel coding at the transmitter and the reverse mapping at the receiver, which are defined as ENM and DNM(Decrypted Nonlinear Mapping). The system is shown in Fig.1, in which Alice stands for the transmitter, Bob for the legitimate receiver and Eve for the eavesdropper.

At the transmitter, an information sequence $\boldsymbol{P}$ is first encoded into a vector $\boldsymbol{I}$ by a channel coder. To handle the burst errors in the output of DNM due to its avalanche effect and improve Bob's performance, we employ a (n, k) systematic linear block code defined over $GF(2^m)$ field, which is able to correct at most t burst errors in a block. Consequently, the block lengths of $\boldsymbol{P}$ and $\boldsymbol{I}$ are k and n symbols in $GF(2^m)$ field, or km and nm bits.

After that, codeword $\boldsymbol{I}$ is mapped with a secret key $\boldsymbol{K}$ into $\boldsymbol{O}$ by ENM before transmitted into the channels. Without loss of generality, we define the lengths

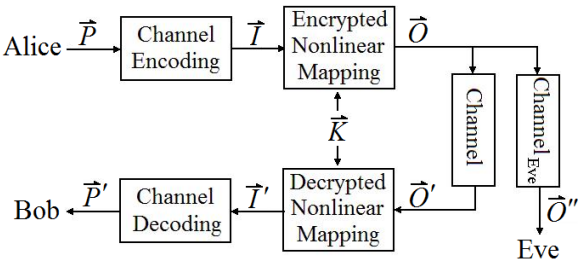


Fig. 1. System Architecture

of $\boldsymbol{K}$ and $\boldsymbol{O}$ as $L = nm$ in bit units, which are equal to the length of codeword $\boldsymbol{I}$. The structure of ENM in this paper can be written as

$$O_i = S\left(I_i\right) \oplus K_i, i = 1, 2, \cdots, n_{ENM}, \tag{1}$$

where $S\left(\cdot\right)$ denotes a nonlinear mapping, which maps I_i to an l-bit vector, and the vector is then Xored with K_i to get O_i. Vectors I_i, K_i and O_i all consist of l bits. Selecting appropriate l making $L \mod l = 0$, that means a codeword includes integer number of ENM input symbols, we can write that $\boldsymbol{I} = [I_1, I_2, \cdots, I_{n_{ENM}}]$, $\boldsymbol{K} = [K_1, K_2, \cdots, K_{n_{ENM}}]$ and $\boldsymbol{O} = [O_1, O_2, \cdots, O_{n_{ENM}}]$, where $n_{ENM} = L/l$ denotes the number of ENMs in a group. The encryption operation given in (1) actually falls into a more general model, namely, additive-like instantaneous block (ALIB) cipher, introduced in [11]. Therefore without specifying a particular nonlinear mapping operation of $S()$ in (1), the cryptanalysis framework established in the following has a fairly extensive scope.

It is notable that $\boldsymbol{I}$ is both the codeword of channel encoding and the input of ENM, but the calculations in channel coding are based on symbols in $GF\left(2^m\right)$ field, while the calculations in ENM are bitwise and based on l-bit to l-bit mappings.

Then ciphertext $\boldsymbol{O}$ is transmitted over two channels, the "Channel" to Bob and "Channel$_{Eve}$" to Eve. Since Bob also possesses the same secret key $\boldsymbol{K}$ as Alice, it implements an inverse operation and derives $\boldsymbol{P'}$, with $\boldsymbol{P'} = \boldsymbol{P}$ when channel decoding corrects all the errors in $\boldsymbol{I'}$.

For Eve, she launches a cryptanalysis in an effort to find out all key bits based on the information she collects. In this paper, secrecy is measured by how hard it is for Eve to get all bits of $\boldsymbol{K}$. Define an ciphertext-only attack as the attack with the ciphertext intercepted from "Channel$_{Eve}$", and without any information about the plaintext. It is notable that with a linear block systematic coding, the linear correlations in the codewords facilitate Eve by providing a linear cryptanalysis method present in an ciphertext-only attack.

2.2 Metrics

Consider an attack strategy used by Eve. In the first stage, she conducts a linear cryptanalysis in an ciphertext-only attack. Because this stage is much faster than

the following brute force attack, she tries to obtain as many bits in the key as possible in order to reduce the time complexity. Define x is the number of bits in $\boldsymbol{K}$ she obtained in this stage, and N is the least number of bits in the ciphertext she needs in this stage. In the second stage, she conducts a brute force attack to get the remaining bits of $\boldsymbol{K}$ with time complexity $T = 2^{L-x}$, where L is the length of $\boldsymbol{K}$. In this stage, she needs at least a pair of ciphertext and plaintext. Correspondingly, two secrecy metrics are defined below.

Ciphertext Amount: defined as the amount of received ciphertext $\boldsymbol{O}''$ in bit units used by Eve to get all bits of $\boldsymbol{K}$. As mentioned above, in the brute force attack stage, the amount of ciphertext needed by Eve is much less than in the linear cryptanalysis stage. Thus, we ignore this part of ciphertext, and use N instead as one of the secrecy metrics in this paper.

Time Complexity: defined as the amount of ciphering operations in the brute force attack. Brute force attack requires calculating the reverse mappings with every possible values of $\boldsymbol{K}$ until the specific ciphertext results in the corresponding plaintext, so the time complexity is $T = 2^{L_K}$, where L_K denotes the amount of secret key bits to search.

The increase on these two secrecy metrics both indicate secrecy enhancement. Besides, a performance metric is also defined below.

Codeword Correction Probability: defined as the probability that a codeword is correctly decoded. Since the avalanche effect of DNM leads to error propagation in the codewords, we use this metric to evaluate the degradation of system's performance, as compared with the case without errors in ciphertexts.

3 Linear Cryptanalysis and Security Metrics

In this Section, we firstly formulate the linear cryptanalysis method, assuming that "Channel$_{Eve}$" is error-free, which means $\boldsymbol{O}'' = \boldsymbol{O}$. Then, the ciphertext amount and time complexity are calculated. The impact of channel errors will be investigated in next section.

3.1 Linear Cryptanalysis

Considering $\boldsymbol{O}'' = \boldsymbol{O}$, Eve can obtain some bits of $\boldsymbol{K}$ from received ciphertext $\boldsymbol{O}$ by a linear cryptanalysis, we will explain how she can do this below. If all the bits in ciphertext $\boldsymbol{O}$ are independent, Eve has to try all the possible values of $\boldsymbol{K}$ until she finds the right one. However, when some part of the ciphertext are interdependent, Eve derives the equations of them, which contain some of the key bits. Then, she can obtain these key bits by solving the equations with a minimum number of ciphertext, which is defined as ciphertext amount in the previous section. In this system, Eve first find the equations about $\boldsymbol{I}$, then convert them to the equations about $\boldsymbol{O}$ and $\boldsymbol{K}$. Taking into account the nonlinear mapping in ENM, Eve approximates it to a linear mapping in order to improve efficiency. Now we describe the linear cryptanalysis method of this system.

Firstly, the linear relationships in $\boldsymbol{I}$ are investigated. The generator matrix of (n,k) linear block systematic codes in $GF(2^m)$ field can be written as $\mathbf{G} = \left[\mathbf{I}_k \vdots \mathbf{Q}\right]$, where $\mathbf{I}_k$ is a identity matrix of size k, and $\mathbf{Q}$ is a $k \times (n-k)$ matrix. Let h^j denote the number of nonzero symbols in the j-th column of $\mathbf{Q}$, and $i_s, s = 1, \cdots, h^j$ denote the indexes of these symbols, thus $q_{i_s}^j$ denotes the s-th nonzero symbol in the j-th column of $\mathbf{Q}$. The codeword $\boldsymbol{r}$ can be written as $\boldsymbol{r} = [r_1, r_2, \cdots, r_n]$, where $r_i, i = 1, \cdots, n$ are symbols in $GF(2^m)$ field, which satisfy the following equations according to coding theory.

$$r_{k+j} = \sum\nolimits_{s=1}^{h^j} q_{i_s}^j \cdot r_{i_s}, j = 1, 2, \cdots, n-k, \tag{2}$$

where r_{k+j} and r_{i_s} denote the $(k+j)$-th and i_s-th symbols in the codeword $\boldsymbol{r}$.

As mentioned in the previous section, the calculations in channel coding and ENM are based on different fields, thus we have to unify them. We give the definitions of "active bit" and "active symbol" next, which will be used several times in the sequel.

Definition (Active Bit): An active bit is the bit which is involved in the linear cryptanalysis.

Definition (Active Symbol): An active symbol is the symbol which contains at least one active bit.

In the j-th equation of (2), r_{i_s} is multiplied by $q_{i_s}^j$ in $GF(2^m)$ field. Only a subset of bits in r_{i_s}, which are determined by $q_{i_s}^j$, are reflected in the result of the j-th equation in (2). We define an m-bit vector γ_i^j as a mask variable, the positions of '1's in which indicate the positions of active bits in r_i in the j-th equation. In order to get the precise expression about γ_i^j and q_i^j, we have to analyze the calculation in the $GF(2^m)$ field to find the relations among all input bits and output bits of the multiplications in (2). As our primary objective of this paper is to establish a framework of cryptanalysis, we will carry γ_i^j throughput the remaining parts without giving a specification on them, thereby making the cryptanalysis framework more general. We know that γ_i^j is active when $i = i_s, s = 1, 2, \cdots, h^j$ or $i = k+j$, while γ_i^j is inactive when $i \neq i_s$ and $i \neq k+j$. Using these mask variables, we can rewrite (2) in bit units as

$$\oplus_{i=1}^{n} \oplus_{t=1}^{m} \left(r_i^j \& \gamma_i^j\right)^t = 0, j = 1, 2, \cdots, n-k, \tag{3}$$

where & denotes bitwise AND operation, and superscript t denotes the t-th bit, thus $\left(r_i^j \& \gamma_i^j\right)^t$ denotes the t-th bit in the result of $r_i^j \& \gamma_i^j$.

Considering $L = mn$ in this system, we have $\boldsymbol{r} = \boldsymbol{I}$, thus can rewrite equation (3) as an equation about $\boldsymbol{I}$. However, when m and l are mismatched, a codeword symbol r_i is not corresponding to an ENM input bits group I_i. In order to indicate the positions of active bits in I_i in the j-th equation, we introduce an l-bit mask variable α_i^j in the following equations.

$$\oplus_{i=1}^{n_{ENM}} \oplus_{t=1}^{l} \left(I_i \& \alpha_i^j\right)^t = 0, j = 1, 2, \cdots, n-k, \tag{4}$$

where $\left(I_i \& \alpha_i^j\right)^t$ denotes the t-th bit in the result of $I_i \& \alpha_i^j$. Let w^j denote the number of active symbols in $\alpha_i^j, i = 1, 2, \cdots, n_{ENM}$, and $i_c, c = 1, \cdots, w^j$ denote their indexes.

It is notable that (3) and (4) actually describe the same linear relationships, but are defined in different fields. Until now, we have the equations about $\boldsymbol{I}$. By combining (1) and (4), we derive a group of nonlinear equations about $\boldsymbol{O}$ and $\boldsymbol{K}$, thus we can get a number of bits in $\boldsymbol{K}$ with sufficient amount of ciphertext. Replacing a nonlinear mapping with a linear operation which holds with certain probability is quite a powerful way in cryptanalysis, namely, linear cryptanalysis [12]. The linear cryptanalysis by Eve can be characterized by the following linear equations, which capture the linear relationship reflected in coded symbols and respective linear approximation for each involved non-linear mapping,

$$\oplus_{t=1}^{l} \left(I_i \& \alpha_i^j\right)^t = \oplus_{t=1}^{l} \left(\left(O_i \oplus K_i\right) \& \beta_i^j\right)^t, \tag{5}$$

where α_i^j is same as in (4), β_i^j is the mask variable of O_i and K_i in the j-th equation, which is also an l-bit vector.

Define $p\left(\alpha_i^j, \beta_i^j\right)$ and $\varepsilon\left(\alpha_i^j, \beta_i^j\right)$ as the probability that (5) holds and the corresponding bias with mask variables α_i^j and β_i^j, where $\varepsilon\left(\alpha_i^j, \beta_i^j\right) \triangleq |p\left(\alpha_i^j, \beta_i^j\right) - \frac{1}{2}|$. The probability $p\left(\alpha_i^j, \beta_i^j\right)$ is obtained by examining how many input and output l-bit pairs can satisfy the linear relationship given in (5) among all possible 2^l input combinations [12].

According to linear cryptanalysis theory [12], an equation with a larger bias requires less amount of ciphertext in the linear cryptanalysis. It is notable that, the linear approximation of nonlinear mappings in this paper is different from the traditional method. Instead of choosing the mask variables α_i^j and β_i^j to maximize the bias $\varepsilon\left(\alpha_i^j, \beta_i^j\right)$, we choose the specific β_i^j which maximize $\varepsilon\left(\alpha_i^j, \beta_i^j\right)$ under the constraint of α_i^j, which is determined by the parameters of channel coding. Consequently, the maximal bias under the constraint of α_i^j is defined as

$$\varepsilon\left(\alpha_i^j\right) = \varepsilon\left(\alpha_i^j, \hat{\beta}_i^j\right) = \max_{\beta_i^j} |p\left(\alpha_i^j, \beta_i^j\right) - \frac{1}{2}|, \tag{6}$$

where $\hat{\beta}_i^j$ is the specific mask variable which maximizes $\varepsilon\left(\alpha_i^j, \beta_i^j\right)$ under the constraint of α_i^j. By combining (4) and (5), we derive the following equations, which include only $\boldsymbol{O}$ and $\boldsymbol{K}$

$$\oplus_{i=1}^{n_{ENM}} \oplus_{t=1}^{l} \left(\left(O_i \oplus K_i\right) \& \hat{\beta}_i^j\right)^t = 0, j = 1, 2, \cdots, n-k, \tag{7}$$

where $\hat{\beta}_i^j$ is used to achieve the maximal bias of linear approximation on each ENM. According to the Piling-up Lemma [12], the bias of (7) is

$$\varepsilon^j = \frac{1}{2}\prod_{c=1}^{w^j} 2\varepsilon\left(\alpha_{i_c}^j\right), j = 1, 2, \cdots, n-k, \tag{8}$$

We have described the process of linear cryptanalysis, and formulated the linear approximate equations about $\boldsymbol{O}$ and $\boldsymbol{K}$, which will be used by Eve in the first stage of attack. It is notable that the relation between m and l is important as it is analyzed in the three specific cases above. Before the end of this subsection, we give a further discussion on this issue. It can be seen from (8) that ε^j is determined by the positions of active bits in $\left[\alpha_1^j, \alpha_2^j, \cdots, \alpha_{n_{ENM}}^j\right]$, which is equivalent to the positions of active bits in $\left[\gamma_1^j, \gamma_2^j, \cdots, \gamma_n^j\right]$ since $nm = n_{ENM}l = L$ is considered in this paper. Consequently, when $\gamma_i^j, i = 1, 2, \cdots, n$ in (3) is determined, the relation between m and l affects only the partition of the active bits, which results in the active symbols $\alpha_{i_c}^j, c = 1, 2, \cdots, w^j$ in (8).

In **Case 2**, we consider the case of $l > m$ and $l \mod m = 0$, which means each input symbol of an ENM consists of integer number of codeword symbols. Here we extend this case by removing the condition of $l \mod m = 0$, and propose the modified partition method. When $l \mod m \neq 0$, an l-bit active symbol cannot over an integer number of m-bit active symbols. In this situation, it is possible that the active bits in an m-bit active symbol γ_i^j are divided into two parts, which separately belong to two l-bit active symbols, namely $\alpha_{i_1}^j$ and $\alpha_{i_2}^j$. Since $l > m$, $\alpha_{i_1}^j$ and $\alpha_{i_2}^j$ possibly include active bits in other m-bit active symbol. As a result, w^j satisfies $\lceil \frac{m\left(h^j+1\right)}{l} \rceil \leq w^j \leq \min\left(2\left(h^j+1\right), n_{ENM}\right)$. Similarly, by removing the condition of $m \mod l = 0$ from **Case 3**, we have $\lceil \frac{h^j+1}{2} \rceil \leq w^j \leq \min\left(\frac{m\left(h^j+1\right)}{l}, n_{ENM}\right)$ in the extended case. From the above analysis, we conclude that the linear cryptanalysis method is applicable in all situations when $nm = n_{ENM}l = L$.

Since not all mask variables $\hat{\beta}_i^j, i = 1, 2, \cdots, n_{ENM}, j = 1, 2, \cdots, n-k$ in (7) consist of '1's, only a part of bits in $\boldsymbol{K}$ can be obtained by Eve in this stage. In order to get the other bits in $\boldsymbol{K}$, she has to carry out brute force attack in the next stage. We next compute the required ciphertext amount and time complexity in the attack.

3.2 Ciphertext Amount

In the following two subsections, we consider one of the strategies Eve may adopt in the attack. According to [12], ciphertext amount can be minimized by using the approximate equation of the maximal bias in traditional linear cryptanalysis. In this paper, due to the constraint of channel coding, Eve cannot maximize all the bias of the ENMs' approximate equations. In order to minimize the ciphertext amount in the linear cryptanalysis, she can use the j_{opt}-th equation in (7), which is the equation with the maximal bias, and the corresponding bias is defined as

$$\varepsilon^{j_{opt}} = \max_{1 \leq j \leq n-k} \varepsilon^j. \tag{9}$$

We investigate the minimum amount of ciphertext required in this strategy first. According to linear cryptanalysis theory [12], Eve obtains a subset of bits in $\boldsymbol{K}$ with success probability P_{Suc}, which increases with the increase of ciphertext amount used in the process. We define $N = N^{j_{opt}} = \min_{1 \le j \le n-k} N^j$ as the minimum ciphertext amount used by the j_{opt}-th equation in (7) with P_{Suc}, given by the following equation,

$$N = \frac{g\left(P_{Suc}\right) l}{\left(\varepsilon^{j_{opt}}\right)^2}, \tag{10}$$

where $g\left(P_{Suc}\right)$ is determined by the method Eve uses in the linear cryptanalysis. Since ciphertext amount is defined in bit units, l is multiplied.

3.3 Time Complexity

After getting a set of key bits by the j_{opt}-th equation of (7), which requires the minimal ciphertext amount, Eve conducts brute force attack to derive the remaining part of $\boldsymbol{K}$. The time complexity is related to the number of bits in $\boldsymbol{K}$ derived in the linear cryptanalysis, thus we calculate this number first. Let $\hat{\beta}^{j_{opt}} = \bigcup_{c=1}^{w^{j_{opt}}} \hat{\beta}_{i_c}^{j_{opt}}$, and define $|\hat{\beta}^{j_{opt}}|$ as the number of bits in $\boldsymbol{K}$ Eve derived from the j_{opt}-th equation in (7), thus we have $|\hat{\beta}^{j_{opt}}| = \sum_{c=1}^{w^{j_{opt}}} |\hat{\beta}_{i_c}^{j_{opt}}|$, where $|\hat{\beta}_{i_c}^{j_{opt}}|$ denotes the number of '1's in $\hat{\beta}_{i_c}^{j_{opt}}$. We define T as the time complexity which refers to the number of ciphering operations required to obtain the rest of key bits and is given by the following equation,

$$T = 2^{L - |\hat{\beta}^{j_{opt}}|}. \tag{11}$$

4 Tradeoff between Security Enhancement and Performance Degradation

We calculate the two secrecy metrics in the previous section under the assumption that "Channel$_{Eve}$" is error-free. In this section, by introducing channel errors into "Channel" and "Channel$_{Eve}$", we investigate the tradeoff between secrecy enhancement and performance degradation of the secure wireless communication system.

4.1 Impact of Channel Errors on Security Metrics

We investigate the impact of channel errors on the ciphertext amount required in the attack. Consider that "Channel$_{Eve}$" is a binary symmetric channel (BSC) with bit error rate (BER) P_e. In this situation, $\boldsymbol{O}''$ is possibly different from $\boldsymbol{O}$, thus only a part of the ciphertext intercepted by Eve is correct. Despite its simplicity, BSC channel has been widely used as one of the primary channel models [13] in wireless systems. We don't consider an actual fading channel herein, whose effect can be translated to a remaining bit error probability. Moreover we

can also consider the bit error events as what are intentionally introduced by Alice in order to further its security against an eavesdropper.

Define O_i'' as an l-bit vector received by Eve, which is modified by channel errors from O_i. The probability of $(O_i'')^t = (O_i)^t$ is $1 - P_e$, where $(O_i'')^t$ and $(O_i)^t$ denotes the t-th bits in O_i'' and O_i. According to the definition in the previous section, we have the bias of $(O_i'')^t = (O_i)^t$ as $\frac{1}{2} - P_e$, where $P_e < \frac{1}{2}$ is assumed without loss of generality. There are totally $|\hat{\beta}^{j_{opt}}|$ active bits in the j_{opt}-th equation of (8), thus according to the Piling-up Lemma, the bias in BSC is

$$\tilde{\varepsilon}^{j_{opt}} = \frac{1}{2}(1-2P_e)^{|\hat{\beta}^{j_{opt}}|} \prod_{c=1}^{w^{j_{opt}}} 2\varepsilon\left(\alpha_{i_c}^{j_{opt}}\right). \tag{12}$$

Define $\tilde{N}$ as the ciphertext amount required by Eve in the BSC, which is given the following equation.

$$\tilde{N} = \frac{g(P_{Suc})\, l}{\left(\tilde{\varepsilon}^{j_{opt}}\right)^2} = \frac{g(P_{Suc})\, l}{\left(\varepsilon^{j_{opt}}(1-2P_e)^{|\hat{\beta}^{j_{opt}}|}\right)^2}. \tag{13}$$

From (10) and (13) we define security enhancement contributed by the BSC channel to Eve as the ratio between the ciphertext amount with BSC and that when its channel is error free, which is given below,

$$\frac{\tilde{N}}{N} = (1-2P_e)^{-2|\hat{\beta}^{j_{opt}}|}. \tag{14}$$

It can be seen from (14) that, $\frac{\tilde{N}}{N}$ increases with the increase of P_e and $|\hat{\beta}^{j_{opt}}|$. Although the time complexity can be increased by decreasing $|\hat{\beta}^{j_{opt}}|$ as mentioned above, $\frac{\tilde{N}}{N}$ sees a decrease at the same time. The tradeoff between this two metrics is a further work we intend to investigate.

4.2 Impact of Channel Errors on Performance Metric

When introducing channel errors into "Channel", the codeword correction probability is affected. Considering "Channel" as a BSC with BER P_e, thus $\boldsymbol{O'} \neq \boldsymbol{O}$, where $\boldsymbol{O}$ and $\boldsymbol{O'}$ are the ciphertext at transmitter and receiver. We evaluate the impact of error propagation on the error correction capability of channel coding. According to the avalanche effect of DNM, as long as at least one bit in O_i' is flipped, one half bits in I_i' are different from I_i. In **Case 1** and **Case 3**, the propagated errors in I_i' belong to a symbol in $GF(2^m)$ field since $m \geq l$. Defining $N_e^{I^{[1,3]}}$ as the number of incorrect symbols in $GF(2^m)$ field in $\boldsymbol{I'}$ in these two cases, we derive its probability cumulative distribution function (CDF).

$$P\left\{N_e^{I^{[1,3]}} \leq x\right\} = \sum_{i=0}^{x} \binom{n}{i} \left(P_e^I\right)^i \left(1-P_e^I\right)^{(n-i)}, 0 \leq x \leq n, \tag{15}$$

where $P_e^I = 1 - (1 - P_e)^m$ denotes the probability that a symbol in $GF(2^m)$ field is incorrect. Consider a channel coding which corrects at most t symbols in $GF(2^m)$ field, the probability that a codeword is correct after channel decoding is $P\left\{N_e^{I[1,3]} \leq t\right\}$, which is the codeword correction probability in **Case 1** and **Case 3**.

Nevertheless, the above result doesn't hold in **Case 2**, since I_i' is a combination of several symbols in $GF(2^m)$ field. As mentioned above, if O_i' contains incorrect bits, a half of bits in I_i' are incorrect. First, the probability that I_i' is incorrect is $\hat{P}_e^I = 1 - (1 - P_e)^l$, where l is the number of bits in I_i'. Then, the number of error bits is $\frac{l}{2}$, and the number of symbols in $GF(2^m)$ field corresponds to the incorrect I_i' is $\frac{l}{m}$ in this case. Since $\frac{l}{m} < \frac{l}{2}$ in general , we approximate that the error bits distribute in all the $\frac{l}{m}$ symbols. Since this is the worst situation, the codeword correction probability in this case is the lower bound of actual value. Define $N_e^{I[2]}$ as the number of incorrect output symbols of DNMs in $\boldsymbol{I'}$ in this case, we derive its CDF.

$$P\left\{N_e^{I[2]} \leq x\right\} = \sum_{i=0}^{\lceil \frac{xm}{l} \rceil} \binom{n_{ENM}}{i} \left(\hat{P}_e^I\right)^i \left(1 - \hat{P}_e^I\right)^{(n_{ENM}-i)}, 0 \leq x \leq n. \tag{16}$$

As a result, $P\left\{N_e^{I[2]} \leq t\right\}$ is the codeword correction probability in **Case 2**. It can be seen from (15) and (16) that, $P\left\{N_e^{I[1,3]} \leq t\right\}$ and $P\left\{N_e^{I[2]} \leq t\right\}$ decrease with the increase of BER when $P_e < \frac{1}{2}$. In the following section on numerical results, we use them as the metric of performance degradation introduced by channel errors.

4.3 Numerical Results and Discussion

For simplicity, the cross over probability of the BSC to Bob is assumed to be equal to that to Eve which can be realized by Alice's inserting BSC error events on purpose. To demonstrate the tradeoff between security enhancement and performance degradation in the presence of channel errors which could be intentionally induced by Alice, we provide some numerical results next for some specific cases.

We consider RS(16,8) as the channel coding in the following numerical results, which means $n = 16$, $k = 8$, $m = 8$, and a codeword can be corrected when it contains at most $t = 4$ incorrect symbols. As for ENM/DNM, we set $l = 8, 16, 4$ separately for **Case 1**, **Case 2**, and **Case 3**. In numerical comparison,, we approximate that $|\hat{\beta}^{j_{opt}}| = L$, which means all bits in $\boldsymbol{K}$ are obtained by Eve in the linear cryptanalysis, thus the brute force attack stage is not required. In the following discussion, we'll analyze the effect of this approximation on the numerical result.

Fig.2 shows the relationship between secrecy enhancement and performance degradation. The vertical axis denotes the enhancement of ciphertext amount

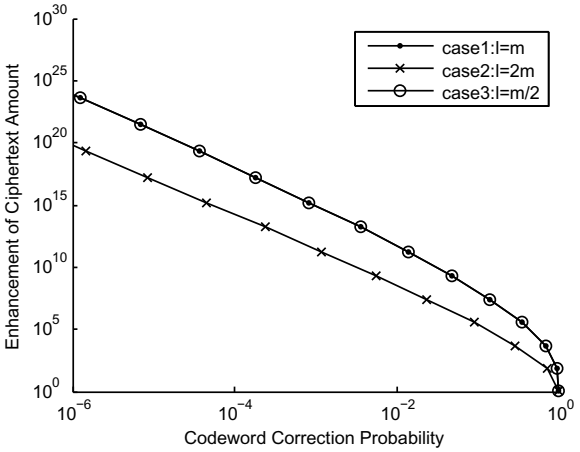


Fig. 2. Simulation Results

and the horizontal axis denotes the probability that a codeword is correct after decoding, which means we compare Bob's performance before and after introducing channel errors by Alice. Therefore in the numerical results the crossover probabilities to Bob and Eve are the same.

It can be concluded from Fig.2 that enhancement of ciphertext amount decreases with the increase of codeword correction probability, which means we can improve secrecy with a degree of performance degradation. Moreover, Fig.2 also shows that performance degradation is higher with a larger DNM output size l, which is because the range of error propagation is larger and more symbols in a codeword are likely to be affected. The lines of **Case 1** and **Case 3** coincide because in these two cases error propagation occurs within a symbol, and doesn't lead to more erroneous symbols in a codeword.

It is notable that, in order to achieve dramatic enhancement on ciphertext amount, the degradation of performance is serious according to Fig.2 We comment that this is because we use the same BER for "Channel" and "Channel$_{Eve}$" in this paper. The enhancement on ciphertext amount is related with the BER of "Channel$_{Eve}$", while the codeword correction probability is related with the BER of "Channel". Under the assumption that "Channel$_{Eve}$" is worse than "Channel", that means the BER is higher in "Channel$_{Eve}$", we will study whether we can achieve more enhancement on ciphertext amount with lower degradation of performance than Fig.2 shows in further research.

5 Conclusion

In this paper, we establish a framework for linear cryptanalysis of the system based on ciphertext-only attack, under which the eavesdropper can explore the inherent linear relationship in the ciphertext and get information of the key.

We investigate the influence of system parameters on the amount of ciphertext and time complexity required in the attack. Furthermore, we make use of the above framework in the presence of channel errors, and evaluate the amount of additional ciphertext required by the eavesdropper and the codeword error probability after decryption and decoding at the legitimate receiver. The tradeoff analysis shows that channel errors give rise to security enhancement with some degree of performance degradation. Besides, it also reveals that channel coding and subsequent block ciphering affect the capability in privacy protection and error correction in a coupled manner.

Acknowledgment. This work is supported by the Fundamental Research Funds for the Central Universities (No.11QG13 and No.12ZP09) and the National Natural Science Foundation of China (No.61105052 and No.71101048).

References

1. Shannon, C.E.: Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication. Bell Systems Technical Journal 28, 656–715 (1949)
2. David, J., Jesse, W.: Overview of IEEE 802.16 Security. IEEE Security and Privacy 2, 40–48 (2004)
3. Zuquete, A., Barros, J.: Physical-layer encryption with stream ciphers. In: IEEE International Symposium on Information Theory, ISIT 2008, pp. 106–110 (2008)
4. Mihaljević, M.J., Imai, H.: An approach for stream ciphers design based on joint computing over random and secret data. Computing 85, 153–168 (2009)
5. Barkan, E., Biham, E., Keller, N.: Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication. J. Cryptol. 21, 392–429 (2008)
6. Basar, K., Levent, E.: Evaluation of GSM Security. In: Proc. of the 5th Symp. on Com. Networks (BAS 2000), pp. 69–78 (2000)
7. Xiao, Y., Chen, H.H., Du, X., Guizani, M.: Stream-based cipher feedback mode in wireless error channel. IEEE Transactions on Wireless Communications 8, 622–626 (2009)
8. Haleem, M.A., Mathur, C.N., Chandramouli, R., Subbalakshmi, K.P.: Opportunistic encryption: A trade-off between security and throughput in wireless networks. IEEE Transactions on Dependable and Secure Computing 4, 313–324 (2007)
9. Imai, H., Hanaoka, G., Maurer, U., Zheng, Y.: Special issue on information theoretic security. IEEE Transactions on Information Theory 52, 4348 (2006)
10. Wyner, A.D.: The wire-tap channel. Bell System Technical Journal 54, 1355–1387 (1975)
11. Lu, S.C.: Random ciphering bounds on a class of secrecy systems and discrete message sources. IEEE Transactions on Information Theory 25, 405–414 (1979)
12. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
13. Cover, T.M., Thomas, J.A., Wiley, J.: Elements of information theory. Wiley Online Library 1, 405–414 (1991)

Differential Privacy Data Release through Adding Noise on Average Value

Xilin Zhang, Yingjie Wu, and Xiaodong Wang

College of Mathematics and Computer Science, Fuzhou University, Fuzhou, China
zhang.xilin@foxmail.com, {yjwu,wangxd}@fzu.edu.cn

Abstract. In recent years, differential privacy data publishing has received considerable attention. However, existing techniques on achieving differential privacy for answering range-count queries fail to release data with high quality. In this paper, we propose a new solution for answering range-count queries under the framework of ε-differential privacy, which aims to maintain high data utility while protecting individual privacy. The key idea of the proposed solution is to add noise on an average tree, in which each node value is the average value of all its leaf nodes. Experimental analysis is designed by comparing the proposed solution and the classic algorithms on the released data utility. The theoretical analysis and experimental results show that our solution is effective and feasible.

Keywords: Differential Privacy, Data Publishing, Adding Noise, Average Tree.

1 Introduction

With the development of science and technology, privacy preserving data publishing [1] becomes an increasingly important research topic. Many organizations (e.g., hospitals, supermarkets, and Census Bureau) collect large volumes of sensitive personal data. Releasing these data to public access may not only greatly benefit scientific research, but also increase the risk to individual privacy at the same time. Methods for publishing useful information while protecting individual privacy have been a research direction in recent years.

In order to publically release these sensitive data with privacy guarantee, many privacy models have been introduced. The well-known k-anonymity[2] notion is proposed first, but it is generally considered too weak. By the defect of k-anonymity, many other notions such as l-diversity[3] and t-closeness[4] are introduced. Unfortunately, these privacy principles are either generally considered as lacking formal mathematical proof, or applicable to limited scenarios.

Dwork[5–8] proposes a privacy model called ε-differential privacy with strong mathematical background comparing with previous privacy models. It represents a major breakthrough in privacy preserving data publishing and changes the research landscape. ε-differential privacy ensures that the removal or addition of a single data record doesn't significantly affect the final released data. And in general framework of differential privacy, the data owner publishes the data through

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 417–429, 2012.

a statistical database and only particular aggregate queries are permitted. The goal is to preserve the privacy of every individual in the database even though the adversaries have auxiliary information about this database already.

Nowadays, differential privacy is widely accepted as one of the strongest privacy guarantees. An effective approach to achieve ε-differential privacy is output perturbation[9] which works by injecting a small random noise. The noise is a random variable with a zero mean. To be specific, the users request a set number of queries and the data owner answers a noisy answer for each query in the interactive model.

In this paper, we work on the problem of answering range-count queries by the framework of differential privacy. The method of injecting noise can well protect the privacy for this problem. But however, the repeated addition of noise will significantly affect the quality of released data. This paper aims to solve this problem by proposing an improved ε-differential privacy model called "DPAV", which achieves differential privacy through adding noise on the average value. DPAV not only can be used for single dimensional range-count queries, but also works well on multidimensional range-count queries. We introduce DPAV on the background of single dimensional first in this paper. And we demonstrate our approach yields prediction accuracy by comparing to other algorithms.

2 Related Works

Dwork[5–8] has issued a series of literatures on privacy preserving data releasing on the notion of differential privacy. Most researches on differential privacy are for answering statistical queries and the primary approach of achieving differential privacy is based on adding Laplace noise to the original results.

We review some major work with the framework of differential privacy here. Xiao[10] developed a technique enforced ε-differential privacy with economical cost and incorporated a query relaxation mechanism. Machanavajjhala[11] used differential privacy to publish the information of the map. Both Korolova[12] and Gotz[13] developed solutions for releasing search logs, the latter achieved differential privacy by (ε,δ)-probabilistic differential privacy and (ε,δ)-indistinguishable. Jagannathan[14] and Friedman[15] proposed techniques for data mining, they figured out the decision tree algorithm with the framework of differential privacy. Xiao[16] introduced a differential privacy algorithm for computing answers with reduced relative errors.

Closely related to our work, there is literature on the problem for range-count queries. Xiao[17] developed a differentially private data release algorithm called Privelet, this algorithm is achieved by wavelet transforms. And Hay[18] proposed a method to generate differentially private histograms for single dimensional range-count queries through a consistency check technique.

Based on the previous works, we focus on range-count queries and provide an approach DPAV. We use a new method to add noise and achieve DPAV. The method provides a formal privacy and utility guarantee for a set of range-count queries.

3 Preliminaries

In this section, we give the background of the problem and detailed definition of ε-differential privacy in 3.1. And we will describe Dwork's basic algorithm[5] and some necessary notions to understand this algorithm in 3.2. We will also point out the defect of Dwork's algorithm in this section.

3.1 Problem Defined

Assume there is a relational table T consists of n elements. The relational table T contains ordered attribute A and there are m possible values of attribute A. Users are allowed to use T by some query sequences. Each sentence of the query sequence is a simple range-count query on a range of attribute A as follows:

$$Select\ count(*)\ from\ T\ where\ T.a \in [x, y]$$

It is obvious that T can be represented by a one-dimensional frequency matrix M with m entries. The i-th ($i \in [1, m]$) entry of M represents the count of attribute A's i-th value. So that users can be answered any range-count query by summing up the entries on M.

Table 1. Relational Table

Name	Age
Alice	30
Bob	30
Jim	32
Lucy	32
Lily	34

Table 2. Frequency Matrix

Age	Count
30	2
31	0
32	2
33	0
34	1

Example 1 (Range-count queries). Assuming Table 1 is table T, so that $n = 5$ and the attribute A is Age. Assuming the range of Age is [30,34] so that $m = 5$. Now we can have a range-count query such as:

$$Select\ count(*)\ from\ T\ where\ T.a \in [32, 34]$$

And Table 2 is M, the frequency matrix of T. It is easy to answer the above query and get the result 3 by summing up the count of Age 32, 33 and 34.

We aim to use an algorithm to answer queries about T with satisfying the requirement of ε-differential privacy.

Definition 1. *(ε-differential privacy[5]) A randomized algorithm Φ satisfied ε-differential privacy, if and only if (i) for any two tables T_1 and T_2 that differ only in one tuple, and (ii) for any output O of Φ, we have*

$$Pr\{\Phi(T_1) = O\} \leq e^{\varepsilon} \times Pr\{\Phi(T_2) = O\} \quad (1)$$

ε measures the strength of the privacy preserving. And in our setting, the output is M^*, which is a releasing version of M. So we want to achieve an algorithm to structure M^*. We will show Dwork's basic algorithm to accomplish ε-differential privacy in the next section.

3.2 Basic Algorithm

By Dwork's theory [5], we can use the Laplace mechanism[5] to release M under differential privacy. It achieves differential privacy by adding noise to each entry in M independently, where the noise is sampled from the Laplace distribution. A Laplace distribution has a probability function:

$$f(\eta) = \frac{1}{2\lambda} e^{-\frac{|\eta|}{\lambda}} \tag{2}$$

By equation (2), we get that the Laplace distribution has an expectation 0 and a variance $2\lambda^2$. After adding noise, the releasing version frequency matrix M^* is accomplished. If we keep ε constant, the magnitude λ depends on the sensitivity of the query sequences, which is defined as follows.

Definition 2. *(Sensitivity [5])). Let F be a set of functions, such that the output of each function $f \in F$ is a real number. The sensitivity of F is defined as*

$$S(F) = \max_{T_1, T_2} \sum_{f \in F} \mid f(T_1) - f(T_2) \mid \tag{3}$$

where T_1 and T_2 are any two tables that differ in only one tuple.

If F is the function set that maps T to M, F will contain m functions in total and each of them output a real number entry in M respectively. If T_1 and T_2 are any two tables that differ in only one tuple, there exists only two functions guarantee $\mid f(T_1) - f(T_2) \mid = 1$ and all others is $\mid f(T_1) - f(T_2) \mid = 0$, so we can get $S(F) = 2$. With Definition 2, Dwork[5] proved following theorem.

Theorem 1. *([5]) Let F be a set of functions with a sensitivity $S(F)$. Let Φ be an algorithm that adds independent noise to the output of each function in F, such that the noise follows a Laplace distribution with magnitude λ. Then, Φ satisfies $(S(F)/\lambda)$-differential privacy.*

By Theorem 1, Dwork's basic algorithm guarantees $(2/\lambda))$-differential privacy, the magnitude λ measures the privacy strength of the algorithm. A bigger λ will lead to a better privacy preserving but a lower quality of releasing data and vice versa.

After adding noise to each entry in M, we can get the new matrix M^*. Any range-count query should be answered by summing up the entries in M^*. A range-count will cover at most m entries in the worst case. If we want to achieve ε-differential privacy by the basic algorithm with a constant ε, we just need to

set $\lambda = 2/\varepsilon$. By Equation (2), we get that the variance of an answer would be $m \cdot 2\,(2/\varepsilon)^2 = 8m/\varepsilon^2$ at most.

When m is huge, the noise variance may be very big for large range so that results in a low utility. That is the motivation for us to present a new algorithm called DPAV. Actually, many literatures (such as [17, 18]) have proposed some solutions to handle the quality problem already. However, DPAV has a better utility and wider applicability than these existing methods.

4 The DPAV Algorithm

In this section, we give some definitions and bring in the process of DPAV in 4.1. In 4.2, we calculate the time complexity first, and then we analyse the answer's noise variance. Besides, we compare DPAV's time complexity and noise variance with Dwork's basic algorithm[5] and Xiao's Privelet[17].

4.1 Algorithm Description

Just as the name of this paper implies, the main difference between basic algorithm and DPAV is on the way of adding noise. And we have described the process by three main steps as Figure 1.

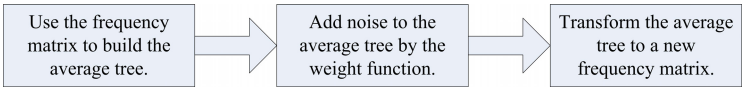


Fig. 1. The process of DPAV

First, we will build a tree B height of $l+1$ (the choice for l's value is discussed in Section 4.2). Before that we need to find the smallest positive integer k to meet the condition $k^l \geq m$. After getting the value of k, we can build the tree B with each internal node containing k children. We just named B an average tree. It is obvious B has $\left(k^{l+1}-1\right)/\left(k-1\right)$ nodes in total.

As B has k^l leaves, we set the first m leaves equate with the m entries in M, and the left $k^l - m$ leaves just initialize to zero. Then we make every internal node equating to the average value of all its leaves, so that every node in B has been assigned a value.

Second, we would add noise to the average tree B. Before analysing how to add noise, we give the definition of generalized sensitivity by Definition 3 and introduce Lemma 1 to connect the generalized sensitivity with ε-differential privacy.

Definition 3. *(Generalized Sensitivity[17]) Let F be a set of functions, each of which takes as input a matrix and outputs a real number. Let W be a function*

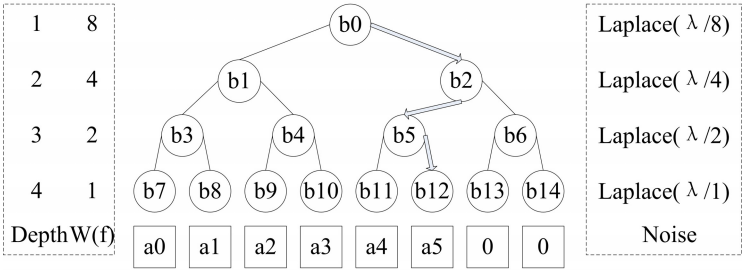


Fig. 2. Average tree

that assigns a weight to each function $f \in F$. The generalized sensitivity of F with respect to W is defined as the smallest number ρ such that

$$\sum_{f \in F} (W(f) \cdot \mid f(M) - f(M') \mid) \leq \rho \cdot \parallel M - M' \parallel_1$$

where M and M' are any two matrices that differ in only one entry, and $\parallel M - M' \parallel_1 = \sum_{v \in M - M'} \mid v \mid$ is the L_1 distance between M and M'.

Lemma 1. *([17]). Let F be a set of functions that has a generalized sensitivity ρ with respect to a weight function W. Let Φ be a randomized algorithm that takes as input a table T and outputs a set $\{f(M) + \eta(f) \mid f \in F\}$ of real numbers, where M is the frequency matrix of T, and $\eta(f)$ is a random variable that follows a Laplace distribution with magnitude $\lambda / W(f)$. Then, Φ satisfies $(2\eta/\lambda)$-differential privacy.*

According to Definition 3, assume that each function of F takes the frequency matrix M as input and outputs a real number, each of the real number is a node in B respectively. Then we define $W(b) = k^{l-i+1}$ which i represents the depth. As shown in Figure 2, $W(b_0) = 8$, $W(b_2) = 4$, $W(b_5) = 2$, $W(b_{12}) = 1$. Then the following Lemma is obtained.

Lemma 2. *The average tree B has a generalized sensitivity of $l + 1$ with respecting to the weight function $W(b) = k^{l-i+1}$.*

Proof. If we change an entry in M to M' by increasing or decreasing a constant δ, then $l + 1$ nodes in B will be changed. One is the corresponding leaf of this entry and others are the ancestors of this leaf. Each of this $l+1$ nodes is changed δ / k^{l-i+1} . Then we can get the generalized sensitivity ρ as follow:

$$\sum_{i=1}^{l+1} \left(k^{l-i+1} \left(\delta / k^{l-i+1} \right) \right) / \delta = l + 1$$

With Lemma 1 and Lemma 2, we see that if we add a Laplace noise with magnitude λ/k^{l-i+1} to each node in the average tree B, we may achieve a $(2(l+1)/\lambda)$-differential privacy. On the other hand, if we want to achieve the ε-differential privacy, we need to add Laplace noise to each node in B with the magnitude $2(l+1)/\left(\varepsilon k^{l-i+1}\right)$.

Third, after the operation of the second step, we get an average tree B with noise. Now, the main job is to transform it to a new frequency matrix M'. For each entry v_x in M^*, we use Equation (4) as follow to construct it.

$$v_x = root + \sum_{i=1}^{l}\sum_{j=1}^{k} g_{ij} b_{ij} \tag{4}$$

Algorithm 1. DPAV algorithm

Input: T, ε
Output: M^*

1. Map the original relational table T to a frequency matrix M that containing m items.
2. Scan l from 1 to $\lceil \log_2 m \rceil$, find l to minimize Equation (5). Let $k = \lceil m^{\frac{1}{l}} \rceil$ and $\lambda = 2(l+1)/\varepsilon$.
3. Build an average tree B with high $l+1$ and every inside node has k sons.
4. Set the first m leaves equal to the items of M and set other leaves to zero.
5. Set every inside node's value equal to the average value of its k sons with a bottom-up order.
6. Add $Laplace\left(\lambda/k^{l-i+1}\right)$ to every node, i represents the depth.
7. Use the dynamic programming equation to compute the value $dp(b)$ of every node with a top-down order.
8. Set M^* equal to the values $dp(b)$ of all the leaves in order and return M^*.

In Equation (4), *root* represents B's root. Every entry v_x has a corresponding leaf $leaf_x$. As each $leaf_x$ has l ancestors totally, b_{ij} is the jth son of the ancestor with depth i, and g_{ij} is as follows.

$$g_{ij} = \begin{cases} 1 - 1/k & The\ jth\ son\ is\ the\ ancestor\ of\ leaf_x \\ -1/k & The\ jth\ son\ isn't\ the\ ancestor\ of\ leaf_x \end{cases}$$

Example 2. Figure 2 is an average tree with $l = 3, k = 2, m = 8$. The leaf b_{12} satisfies $b_{12} = a_5$. The internal node b_2 satisfies $b_2 = (b_{11} + b_{12} + b_{13} + b_{14})/4$. Other nodes are similar.

As the depth showed in the left in Figure 2, there are the noises should be added in the right. After adding noise, assuming we want to calculate a_5^*, the corresponding leaf of a_5^* is b_{12}, the ancestors of b_{12} is b_0, b_2, b_5, so a_5^* can be constructed by:

$$a_5^* = root + \left(1 - \frac{1}{k}\right) b_2 - \frac{1}{k} b_1 + \left(1 - \frac{1}{k}\right) b_5 - \frac{1}{k} b_6 + \left(1 - \frac{1}{k}\right) b_{12} - \frac{1}{k} b_{11}$$

Other entry can be constructed as the same method.

After adding noise, it seems taking $O(mkl)$ time complexity to construct M^* by B. But exactly, it can optimization to $O(m)$ by a top-down scan with a dynamic programming algorithm as many computation steps are repeating. The dynamic programming equation is:

$$\begin{cases} dp(b) = root & When\ b\ is\ the\ root \\ dp(b) = dp(b_{father}) + b - \sum_{j=1}^{k} \frac{b_j}{k} & When\ b\ isn't\ the\ root \end{cases}$$

b_{father} is b's father, $b_1, b_2, ..., b_k$ is b_{father}'s k sons. After a top-down scanning by the dynamic programming equation, every node b gets a value $dp(b)$. And the values $dp(b)$ of all the leaves compose M^*. Algorithm 1 gives the detail process of DPAV.

4.2 Algorithm Analysis

Let's analyse the time complexity first. It takes $O(n+m)$ maping T to M. And it takes $O(m)$ to build B because k^l has a same series with m. The dynamic programming algorithm takes $O(m)$. So the time complexity of DPAV is $O(n+m)$, just the same with Dwork's basic algorithm and Xiao's Privelet[17].

We will point out the upper bound of the range-count query's noise variance next. And we need following lemma first.

Lemma 3. *While achieving DPAV, k sons of a node in B will cause a noise variance $k\sigma^2/2$ at most.*

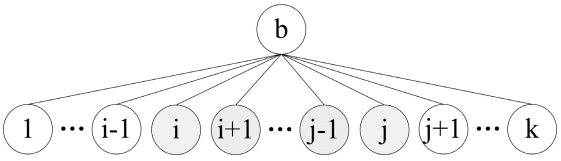


Fig. 3. Node b and its sons

Proof. Assume that a node b in B has a depth h. Every son of node b has cover k^{l-h} leaves totally. Let's take just x continuous sons $b_i, b_{i+1}, ..., b_j$ $(1 < i < j < k)$ to be covered completely, and the sons b_{i-1} and b_{j+1} are covered y $\left(0 \le y \le k^{l-i}\right)$ and z $\left(0 \le z \le k^{l-i}\right)$ sons, respectively. According to Equation (4), we can infer as follow.

The noise variance of b_{i-1}:

$$error(b_{i-1}) = \left(y \cdot \frac{k-1}{k} - x \cdot k^{l-h} \cdot \frac{1}{k} - z \cdot \frac{1}{k}\right)^2 \cdot 2\left(\frac{\sigma}{k^{l-h}}\right)^2$$

The noise variance of b_i to b_j:

$$error(b_i, ..., b_j) = x\left(k^{l-h} - x \cdot k^{l-h} \cdot \frac{1}{k} - y \cdot \frac{1}{k} - z \cdot \frac{1}{k}\right)^2 \cdot 2\left(\frac{\sigma}{k^{l-h}}\right)^2$$

The noise variance of b_{j+1}:

$$error\left(b_{j+1}\right) = \left(z \cdot \frac{k-1}{k} - x \cdot k^{l-h} \cdot \frac{1}{k} - y \cdot \frac{1}{k}\right)^2 \cdot 2\left(\frac{\sigma}{k^{l-h}}\right)^2$$

The noise variance of the remaining sons $b_1, ..., b_{i-2}, b_{j+2}, ..., b_k$:

$$error\left(remaining\right) = (k-x-2)\left(x \cdot k^{l-h} \cdot \frac{1}{k} + y \cdot \frac{1}{k} + z \cdot \frac{1}{k}\right)^2 \cdot 2\left(\frac{\sigma}{k^{l-h}}\right)^2$$

Suppose $y' = \frac{y}{k^{l-h}}, z' = \frac{z}{k^{l-h}}$, stack up and simplify the above four equations:

$$error\left(b\right) = \frac{2\sigma^2}{k}\left[\frac{k^2}{4} - \left(x + \frac{2y' + 2z' - k}{2}\right)^2 - y'\left(1-y'\right)k - z'(1-z')k\right]$$

Finally we can get:

$$error\left(b\right) \leq \frac{2\sigma^2}{k} \cdot \frac{k^2}{4} = \frac{k\sigma^2}{2}$$

The lemma is proved.

By Lemma 3, we can get Lemma 4.

Lemma 4. *If the Laplace magnitude is λ, any range-count query's noise variance of DPAV is at most $(2+kl)\lambda^2$.*

Proof. root will be added at most k^l times as it covers k^l sons, so the noise variance of covering *root* is $2\left(k^l \cdot \frac{\lambda}{k^l}\right) = 2\lambda^2$. For any range-count query, each node will be covered by the three cases:

1. None of its leaves are covered.
2. All of its leaves are covered.
3. A part of its leaves are covered.

The first two cases have a noise variance 0 and can be ignored. And it exists at most 2 nodes of the third case in every depth. By Lemma 3, the total noise variance is:

$$2\lambda^2 + \sum_{i=1}^{l}\left(2 \cdot \frac{k\lambda^2}{2}\right) = (2+kl)\lambda^2$$

By $k = \lceil m^{\frac{1}{l}} \rceil, \lambda = 2\left(l+1\right)/\varepsilon$ and Lemma 4, the noise variance can be written as:

$$\left(2 + \lceil m^{\frac{1}{l}} \rceil l\right)\left(2\left(l+1\right)/\varepsilon\right)^2 \tag{5}$$

If m is known already, we just need to find l to minimize Equation (5). We can get $1 \leq l \leq \lceil \log_2 m \rceil$ as k is over 1. So after scanning l from 1 to $\lceil \log_2 m \rceil$, we can figure out the best choice of l, and then get k .

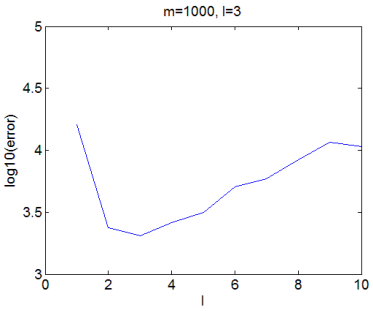


Fig. 4. The change of noise variance varying l

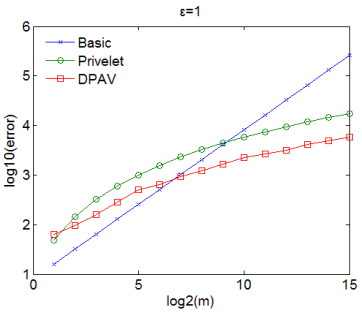


Fig. 5. The noise variance of the algorithms

Figure 4 shows the change of Equation (5) as l when $m = 1000$. We can see it get the smallest value while $l = 3$ and $k = 11$. In other words, every node in B has 11 sons is best choice while $m = 1000$.

For every count-range query, the basic algorithm has a noise variance $8m/\varepsilon^2$, and Xiao's Privelet has a noise variance $(2 + \lceil \log_2 m \rceil)\left(2\left(\lceil \log_2 m \rceil + 1\right)/\varepsilon\right)^2$[17]. Figure 5 shows the change of the noise variance for these three algorithms as m taking $2, 2^2, ..., 2^{15}$. It can come to a conclusion that the noise variance of DPAV is lower than Privelet in most cases by the figure.

5 Experiments

We will show the experimental results of comparing the data utility of basic algorithm, Privelet and DPAV. The experimental platform is Intel Pentium Dual2.70GHz processor and 2GB RAM and it runs in Windows XP. The program is coded by Dev-C++ 4.9.9.2, and the figures are made by MATLAB 7.0. We compare the error with the results by Equation (6):

$$error = \sum_{q \in Q} \left(q\left(M^*\right) - q\left(M\right)\right)^2 \tag{6}$$

We use two datasets[19, 20] to do the experiments. The first dataset[19] is Amazon Access Samples Data Set. It provides the access history from March 1, 2005 to August 31, 2010 of Amazon. It has 716064 records and it contains 2010 days in total so that $m = 2010$. The second dataset[20] is American Community Survey 2006-2010 Acs 5-Year Pums. It contains $n = 3597186$ users and we choose the attribute Income. The range of Income is from -2197 hundred to 15330 hundred, so $m = 17528$.

We do the experiments by takings ε with value 0.5, 0.75, 1 and 1.25. A larger *varepsilon* means lower privacy and fewer noise. We accomplish the experiments with two sets.

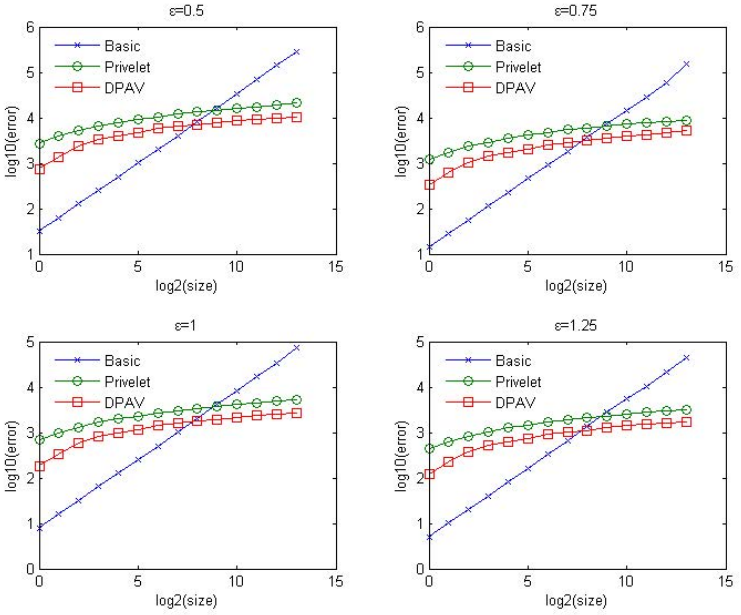


Fig. 6. Experiments of varying range size

In the first set of experiments, we use dataset[20]. We observe the error of range size $2^0, 2^1, ..., 2^{13}$ in turn. In order to make the result as accurate as possible, 1000 random range-count queries are generated and the average error is taken for each range size. Again we repeat the process for 50 times and average the results.

Figure 6 shows the results varying range size. For the sake of observing, we draw the figures by the X axis representing $\log_2(size)$ and Y axis representing $\log_{10}(error)$. We can find that the error of basic algorithm is smaller than Privelet and DPAV for small range. But when the size increases, the error of basic algorithm grows up sharply comparing with the other two algorithms, so error of the basic algorithm is larger than the other two algorithms for large range.

Then we compare DPAV to Privelet. From Figure 6, we can see that no matter what size the range is, the error of DPAV is uniformly lower than Privelet. Though the superiority seems noly a little, but we just have taken the Y axis a logaritmim.

In the second set of experiments, we create 10000 random range-count queries and calculate the total error. Figure 7 and Figure 8 shows the total error of basic algorithm, Privelet and DPAV varying ε of two datasets respectively.

As the noise magnitude decreases with the increase of ε, we can find that a larger ε results in a lower error. The huge error of basic algorithm is emerged thoroughly which indicates its defect. What's more, the error of Privelet is almost two times larger than that of DPAV. DPAV has the lowest error among the three algorithms demonstrate its effectiveness.

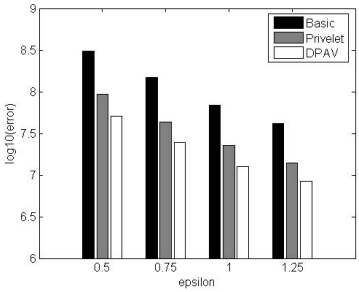


Fig. 7. Experiments of random queries (Access)

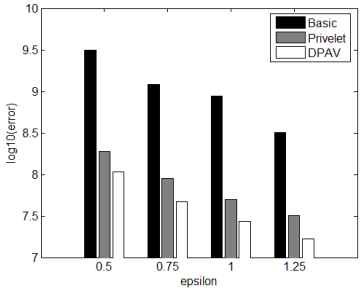


Fig. 8. Experiments of random queries (Income)

In summary, our experiments demonstrate that DPAV has a better data utility than basic algorithm and Privelet, which is consistent as our theory. And specifically, the error of DPAV is even a half of Privelet approximately so that our improved algorithm is significative.

6 Conclusions

In this paper, we study the problem of answering count-range query under the framework of ε-differential privacy. And we present a quite interesting and effective algorithm — DPAV, and analyze the advantages of this algorithm comparing with existing solutions by mathematical proof. Finally, we use the experimental results to demonstrate that DPAV does release high-utility data comparing with several famous existing solutions.

What's more, DPAV can be used to other data type and multi-conditioned query. It is significant to extend DPAV to more complex query. Due to the low noise variance of DPAV, we believe it can keep providing a good utility as well. But by the limitation of the paper's length, it is left as the future work.

References

[1] Fung, B.C.M., Wang, K., Chen, R., Yu, P.S.: Privacy-preserving data publishing: A survey on recent developments. ACM Computing Surveys 42(4) (2010)
[2] Sweeney, L.: k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems 10(5), 557–570 (2002)
[3] Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data 1(1), Article 3 (2007)
[4] Li, N., Li, T., Venkatasubramanian, S.: t-closeness: Privacy beyond k-anonymity and l-diversity. In: IEEE 23rd International Conference on Data Engineering, ICDE 2007, pp. 106–115 (2007)

[5] Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating Noise to Sensitivity in Private Data Analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
[6] Dwork, C.: Differential Privacy. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4052, pp. 1–12. Springer, Heidelberg (2006)
[7] Dwork, C.: Differential Privacy: A Survey of Results. In: Agrawal, M., Du, D.-Z., Duan, Z., Li, A. (eds.) TAMC 2008. LNCS, vol. 4978, pp. 1–19. Springer, Heidelberg (2008)
[8] Dwork, C., Smith, A.: Differential privacy for statistics: what we know and what we want to learn. Journal of Privacy and Confidentiality 1(2), 135–154 (2009)
[9] Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: the sulq framework. In: PODS, Baltimore, Maryland, pp. 128–138 (2005)
[10] Xiao, X., Tao, Y.: Output perturbation with query relaxation. Journal Proceedings of the VLDB Endowment, 857–869 (2008)
[11] Machanavajjhala, A., Kifer, D., Abowd, J.M., Gehrke, J., Vilhuber, L.: Privacy: Theory meets practice on the map. In: IEEE 24th International Conference on Data Engineering, ICDE 2008, pp. 277–286 (2008)
[12] Korolova, A., Kenthapadi, K., Mishra, N., Ntoulas, A.: Releasing search queries and clicks privately. In: WWW (2009)
[13] Gotz, M., Machanavajjhala, A., Wang, G., Xiao, X., Gehrke, J.: Privacy in search logs. CoRR, abs/0904.0682 (2009)
[14] Jagannathan, G., Pillaipakkamnatt, K., Wright, R.N.: A practical differentially private random decision tree classifier. In: IEEE International Conference on Data Mining Workshops, ICDMW 2009, pp. 114–121 (2009)
[15] Friedman, A., Schuster, A.: Data mining with differential privacy. In: KDD 2010, Washington, DC, USA, pp. 493–502 (2010)
[16] Xiao, X., Bender, G., Hay, M., Gehrke, J.: iReduct: differential privacy with reduced relative errors. In: Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data, New York, USA, pp. 229–240 (2011)
[17] Xiao, X., Wang, G., Gehrke, J.: Differential privacy via wavelet transforms. In: 2010 IEEE 26th International Conference on Data Engineering (ICDE), pp. 225–236 (2010)
[18] Hay, M., Rastogi, V., Miklau, G., Suciu, D.: Boosting the accuracy of differentially private histograms through consistency. Proceedings of the VLDB Endowment 3(1), 1021–1032 (2010)
[19] Amazon Access Samples Data Set, `http://archive.ics.uci.edu/ml/index.html`
[20] American Community Survey 2006-2010 Acs 5-Year Pums, `http://www.census.gov/`

Private Friends on a Social Networking Site Operated by an Overly Curious SNP

Roman Schlegel and Duncan S. Wong

Department of Computer Science
City University of Hong Kong
sschlegel2@student.cityu.edu.hk, duncan@cityu.edu.hk

Abstract. These days, social networking sites are more popular than ever, with some sites having dozens or even hundreds of millions of users. At the same time, users on these sites are sharing an unprecedented amount of personal information, generating serious privacy concerns. Personal and sensitive content shared by users on social network sites is barely protected from access by unauthorized users and the Social Networking Provider (SNP) itself always has access to all content. To solve this problem, some existing solutions solicit an external third-party server to provide online privacy protection of content shared by users on social networking sites; other solutions incur a key distribution overhead among the users who are sharing content. These solutions usually have a noticeable impact on the user experience, or are susceptible to single-point-of-failure problems by requiring an external server.

In this paper, we propose a new solution which can achieve the following two desirable features through a novel application of a constant-size-ciphertext broadcast encryption scheme: (1) content posted by a user can only be read by authorized users and nobody else, not even the SNP itself; (2) no key distribution or any external server is necessary during normal operations. Apart from a key extraction server which is contacted only once by each user during an initial registration, the system is self-contained within the web browser (using a plugin) of each user. The system can be used directly with existing social networking sites. We also implemented a prototype for Facebook and perform a thorough evaluation which shows that the scheme is feasible, scalable and practical.

1 Introduction

With the popularity of social networking sites, users are sharing more and more personal information with friends, friends of friends and even strangers. Already in 2005, Gross and Acquisti [1] showed that people disclose a significant amount of personal and even intimate information. The most popular social network nowadays is Facebookwith over 800 million users [2]. Although many social networking sites have privacy settings which allow users to restrict who can access personal information, these settings are not always intuitive and research by boyd et al. [3] suggests that they can be difficult to use. Even though users

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 430–444, 2012.

might be able to restrict access to personal information, by the very nature of it, the Social Networking Provider (SNP) has access to all personal information shared by a user. Furthermore, the SNP might not only make the shared information available to other users, but also use it commercially (e.g., for advertising purposes) outside of the context of the social networking site. Personal information shared with the SNP is also at risk of being released through break-ins of hackers or accidental data leaks [4,5,6].

While these risks always exist for online social networking sites, simply not using them is not a realistic or helpful suggestion. boyd [3] showed that even younger adults, for whom it has been argued that they do not care about privacy, in fact actively take steps to adjust their privacy settings, showing that they do care. Acquisti and Gross [7] also found that privacy is important.

In this paper we propose a system which allows users to share personal information with friends on a social networking site in an efficient, practical and transparent manner, while preserving privacy by preventing access to that information from anybody outside of the circle of friends of a user, including the SNP itself. Existing solutions have a number of drawbacks, they usually require an external server to be online all the time, do not integrate well with existing social networking sites, or have limited performance. Compared to these existing solutions, our system (1) does not need a third party server for normal encryption/decryption operations; (2) has a key extraction server which only needs to be connected to once for each user during registration; (3) integrates seamlessly with existing social networking sites and (4) is completely self-contained within the browser of a user once setup. Our contributions in this paper are:

- we present a new system which uses broadcast encryption to post content on a social networking site such that the content can only be viewed by a user's friends, and nobody else, not even the SNP itself
- we introduce an encoding algorithm to display binary data as UTF-8 text which requires almost 30% less display space than existing encodings
- compared to previous research, our system does not require external third-party servers or key distribution and is self-contained within the browser
- we present and evaluate a prototype implementation for the popular site Facebook, demonstrating that our scheme is practical and efficient

Paper Structure. In Sec. 2 we give an overview of the system and introduce the necessary cryptographic primitives. We then describe our system in more detail in Sec. 3 and proceed to the evaluation of our prototype in Sec. 4. We discuss limitations and future work in Sec. 5, give an overview of related work in Sec. 6 and finally conclude in Sec. 7.

2 Overview

Our system involves two entities (Fig. 1): an SNP (such as Facebook, Google+, Twitter, etc.) and a user, who uses a plugin within a web-browser to realize our

system. This plugin seamlessly encrypts information posted on a social networking site *before* it leaves the user's computer, so that only the user's friends can decrypt it. Also, the plugin decrypts content posted by the user's friends *after* the content is downloaded to the user's computer. The user has full control over which friends are authorized to decrypt the posted content. When compared with a plaintext message, the ciphertext in our system is only 30-50% larger in display size, thanks to a novel application of an efficient Identity-Based Broadcast Encryption (IBBE) and a new encoding method for displaying ciphertext also proposed in this paper. Our system solves some of the problems present in existing solutions, such as key distribution and the requirement for an online, trusted encryption/decryption server (see Sec. 6).

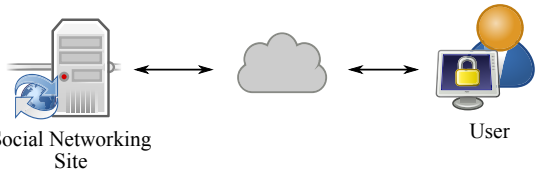


Fig. 1. System Overview: The user, with a cryptographic browser plugin, interacts with a social networking site on the Internet, so that data can be encrypted *before* it is sent to the social networking site

2.1 Identity-Based Broadcast Encryption

Broadcast encryption is a cryptographic primitive which allows to encrypt a message simultaneously for multiple recipients with constant ciphertext size. Moreover, using *Identity-Based* Broadcast Encryption (IBBE) obviates the need for managing user public keys, as the *identity* (e.g. username, user ID, email address, etc.) is used to encrypt a message for a particular user. An efficient IBBE scheme with constant size ciphertexts and private keys is due to Delerablée [8]. In addition, the size of the system parameters (also known as master public key) is linear in the maximal number m of recipients (which is generally much smaller than the number of possible identities in the system).

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups of prime order p, let g_1 a generator of $\mathbb{G}_1$ and g_2 a generator of $\mathbb{G}_2$. $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ is a bilinear pairing where (1) Bilinear: for all $P \in \mathbb{G}_1$, $Q \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, $e(P^a, Q^b) = e(P, Q)^{ab}$; (2) Non-degenerate: $e(g_1, g_2) \neq 1$; and (3) Computable: $e(P, Q)$ can be computed efficiently for all $P \in \mathbb{G}_1$ and $Q \in \mathbb{G}_2$.

Setup. Suppose the maximum number of recipients in each encryption is m. An Key Extraction Server randomly generates two generators g and h for $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively, and $\gamma \leftarrow \mathbb{Z}_p^*$. Let $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p^*$ be a hash function. The master secret key and public key of the Key Extraction Server are $\mathsf{MSK} = \{g, \gamma\}$ and $\mathsf{PK} = \{w, v, h, h^\gamma, \ldots, h^{\gamma^m}\}$, respectively, with $w = g^\gamma$ and $v = e(g, h)$.

Extract. The secret key for an individual user with identity ID is extracted using MSK: $\mathsf{sk}_{\mathsf{ID}} = g^{\frac{1}{\gamma+\mathcal{H}(\mathsf{ID})}}$.

Encryption. To encrypt a message for s recipients ($s \leq m$) with $\mathsf{IDs} = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_s\}$, the broadcaster first randomly picks a $k \leftarrow \mathbb{Z}_p^*$ and then calculates

$$C_1 = w^{-k}, \quad C_2 = h^{k \cdot \prod_{i=1}^{s}(\gamma+\mathcal{H}(\mathsf{ID}_i))}, \text{ and } K = v^k \tag{1}$$

uses K to encrypt the message and includes $\mathsf{Hdr} = (C_1, C_2)$ as the encryption header.

Decryption. Given an encryption header $\mathsf{Hdr} = (C_1, C_2)$, a user with identity ID_i and secret key $\mathsf{sk}_{\mathsf{ID}_i}$, recreates the encryption key K by calculating

$$K = \left(e\left(C_1, h^{p_{i,S}(\gamma)}\right) \cdot e(\mathsf{sk}_{\mathsf{ID}_1}, C_2)\right)^{\frac{1}{\prod_{j=1, j\neq i}^{s} \mathcal{H}(\mathsf{ID}_j)}} \tag{2}$$

where $p_{i,S}(\gamma)$ is calculated as

$$p_{i,S}(\gamma) = \frac{1}{\gamma}\left(\prod_{j=1, j\neq i}^{s}(\gamma + \mathcal{H}(\mathsf{ID}_j)) - \prod_{j=1, j\neq i}^{s} \mathcal{H}(\mathsf{ID}_j)\right). \tag{3}$$

2.2 CJK Encoding for Ciphertexts

The encryption output is in binary format and has to be converted for display and storage, commonly to Base64. One drawback is that each displayed character only contains 6 bits of information, making the number of Base64 symbols 33% larger than the number of bytes.

Because UTF-8 is becoming more common as an encoding standard on the Internet, we developed an algorithm to encode binary data into UTF-8 characters, more specifically into CJK (Chinese, Japanese and Korean) characters.[1] While UTF-8 uses up to 4 bytes to store one character, when *displaying* said character it will only use *one* character space. This encoding allows us to pack 14 bits into each displayed character, instead of just 6 bits for Base64. This means we can save 43% in the number of displayed characters by using our new encoding method rather than expanding the display space by 33% for the case of Base64. The goal here is to save *display space* when displaying ciphertext to minimize the graphical impact of replacing plaintext with ciphertext, *not* to save storage space. Figure 2 shows a comparison of CJK and Base64. Note that the actual saving in display space may not reach 43% as each character in CJK encoding is wider than a conventional ASCII character.

[1] A similar idea has independently been proposed for encoding an image into text: `http://www.flickr.com/photos/quasimondo/3518306770/in/set-72057594062596732/`. In our encoding we address a different problem, namely, how to encode binary data into as few displayable characters as possible.

(a) CJK (b) Base64

Fig. 2. This figure shows how CJK encoding saves approximately 30% in display space when compared with Base64

Algorithm for Encoding Binary Ciphertext Strings. The Unicode standard defines a range for CJK characters at 0x4E00-0x9FFF (the range is slightly over 14 bits long), the so-called *basic CJK Unified Ideographs block*. To use our encoding, the binary data is split into 14-bit blocks, and each block is converted into one CJK character by interpreting its value as an index within the CJK range. The resulting CJK character is then converted to UTF-8 and displayed. Figure 2 shows a comparison of Base64 encoding and CJK encoding on the same binary input, and the display space needed for the CJK encoding is approximately 30% smaller than Base64.

2.3 Threat Model

There are a number of potential adversaries in our system. We review them here and also specify our assumptions.

Social Networking Provider (SNP). One potential adversary is the SNP, which no longer has access to the posted content of its users due to the user-side encryption. A provider could try a number of malicious tricks, for example by adding JavaScript-based key-loggers to its website, but systems exist to mitigate such attacks ([9,10]). An SNP might also archive content posted by its users, secretly keep messages which have been deleted, keep old versions of content and keep all posted content of a user even after a user deletes his profile or account. We consider the SNP to be an *honest-but-curious* player in the system, posting messages submitted by users honestly, but also trying to recover the encrypted posts. Under this model, our system can successfully keep the SNP from learning the plaintext of content posted on its site.

Strangers. In this context, "strangers" refers to people who are also members of the social networking site, but who are not in the circle of friends of a user. Strangers might try to discover personal information about a user, for example by accessing information which is not properly protected by the privacy settings, e.g., a user might have mistakenly set the privacy settings too permissive. As long as a user does not explicitly add a stranger to its circle of friends, the stranger will not be able to access any content protected by our system. In addition, even

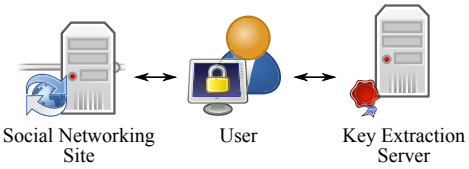


Fig. 3. The system architecture consists of a social networking site, a user running a plugin in the browser on the local computer and a key extraction server which only needs to be contacted once when a user joins the system to get a personal decryption key. During normal operations, no external server is necessary.

within its circle of friends a user can explicitly select which friends should be able to access/decrypt a user's content, providing even more fine-grained control.

Other Assumptions. We assume that neither the computer or web-browser of a user has been compromised and we do not consider social engineering attacks.

3 System Architecture

In this section we will describe the architecture of our system. While our system is general enough that it can be adapted for different social networking sites, we focus on the example of Facebook and describe our system in that context.

3.1 System Components

Our proposed system consists of the following components, shown in Figure 3:

Social Networking Provider. This is the provider of a web-site which allows a user to maintain and interact with a social network. Popular social network service providers are for example Facebook, Google or Twitter. Even if the site offers privacy settings to limit what can be accessed by other users, by necessity the provider itself has access to all information stored by users.

Web-Browser Plugin. To make use of our system, a user simply continues using a web-browser such as Google Chrome, Firefox or Safari, but in addition installs a local plugin, which transparently and unobtrusively enables secure communication on the social networking website as described below. The plugin contains the necessary code for encryption and decryption, encoding and decoding, recipient list management, key extraction and seamless integration.

Key Extraction Server(s). This server will assist each user in obtaining an individual secret key necessary for the encryption scheme (see below). It must be managed by a different entity than the SNP to prevent collusion. Note that this server only needs to be contacted once by each user during the setup stage, and is no longer required during normal operations of the system.

3.2 Encryption Algorithm

The encryption algorithm uses the broadcast encryption scheme described in Section 2.1 as a building block to realize encrypted message exchange on Facebook. The inputs to the encryption algorithm are: (1) the IBBE master public key PK, (2) a message $\mathcal{M}$, and (3) a recipient list $\mathcal{R}_x$ of s Facebook users (friends) with IDs $= \{\mathrm{ID}_1, \ldots, \mathrm{ID}_s\}$, including the current user (Facebook assigns a numerical user ID for each user, these are the IDs used in the recipient list). The steps to encrypt a message $\mathcal{M}$ are as follows, on input $\mathsf{PK}, \mathcal{M}$, IDs,

1. calculate encryption headers C_1, C_2 and instance key K according to Eq. 1
2. compute $K' \leftarrow \mathrm{SHA256}(f(K))$, where $f : \mathbb{G}_T \rightarrow \{0,1\}^*$ maps an element of $\mathbb{G}_T$ to a binary string
3. encrypt $\mathcal{M}$ under K', i.e., $\mathcal{C} \leftarrow \mathrm{AES}_{K'}(\mathcal{M})$
4. encode the identifier x of the recipient list $\mathcal{R}_x$, the ciphertext $\mathcal{C}$ and the encryption headers C_1, C_2 using CJK encoding and form $\mathcal{C}' = \mathrm{CJK}(x)||\ \mathrm{CJK}(C_1||C_2)||\mathrm{CJK}(\mathcal{C})$, then output $\mathcal{C}'$

Note that $\mathcal{C}'$ does not include the actual recipient list $\mathcal{R}_x$, but only an identifier x. As $\mathcal{R}_x$ does not change frequently, it is more economical to store $\mathcal{R}_x$ separately and include the identifier of the relevant recipient list in each message.

3.3 Decryption Algorithm

The input for the decryption algorithm to recover the plaintext message is the ciphertext $\mathcal{C}'$ and the personal decryption key $\mathsf{sk}_{\mathsf{ID}}$ of the user attempting to decrypt the message. The decryption algorithm has the following steps:

1. use CJK decoding to recover $x, C_1, C_2, \mathcal{C}$ from $\mathcal{C}'$
2. use x to retrieve $\mathcal{R}_x$ which is stored separately, if ID of the decrypting user is not in $\mathcal{R}_x$, decryption stops
3. otherwise, compute K according to eqs. 2 and 3
4. compute $K' = \mathrm{SHA256}(f(K))$, decrypt $\mathcal{C}$ as $\mathcal{M} = \mathrm{AES}^{-1}_{K'}(\mathcal{C})$, output $\mathcal{M}$

3.4 Actual Usage

In the following we describe how the system works in the context of Facebook in more detail. As mentioned earlier, the system generalizes easily to other social networking sites, but we chose Facebook as an example because of its popularity.

Setup. The setup process for the user is divided into two easy steps as follows:

1. Install a plugin in the user's web-browser. This can be as easy as clicking a link and confirming the installation.
2. Initialize the plugin by logging into Facebook and authorizing the plugin. The plugin will then automatically contact the key extraction server to retrieve the personal decryption key for the user.

Once the plugin has been installed, it automatically integrates with Facebook in the background by keeping an up-to-date recipient list and transparently encrypting/decrypting posted messages.

As described in Section 3.2, the list of recipients is stored separately from the encrypted messages. Facebook, for example, allows to store notes with arbitrary content, and each note is uniquely identified by an ID. Our plugin therefore stores the current list of recipients in a note (creating a new note whenever the list of friends changes) and only references the ID of the note in the ciphertext.

Encrypting a Message. When Alice wants to post a message on her wall, for example, she simply enters the desired message in the input box, and clicks "Post". Then the following tasks are carried out (Fig. 4).

- When Alice clicks on "Post", the plugin intercepts the click to prevent submitting the cleartext message, and copies it from the input box (1).
- The plugin fetches Alice's friends in the form of a reference to the Facebook note containing the latest list of friends (2), as described earlier.
- Using the latest list of friends as the list of recipients, the plugin encrypts the cleartext message using broadcast encryption (3) as described in Section 2.1, and encodes the ciphertext using CJK encoding (see Section 2.2).
- As the last step, the plugin replaces the cleartext message in the input box with the encoded ciphertext, and initiates the submission of the *encrypted message* to Facebook (4).

Decrypting a Message. Thanks to the integration of the plugin into the web browser of the client, the decryption of encrypted messages happens automatically when loading a Facebook page. The plugin identifies all encrypted messages on a page, determines the recipient list for each message, uses the recipient list and the user private key to decrypt a message and replaces the encrypted message with the plaintext. All of this happens automatically within the browser. The user only sees the plaintext content, making the encryption transparent.

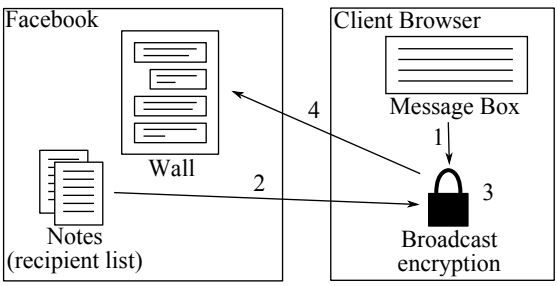


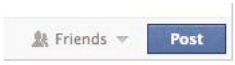
Fig. 4. This figure shows the process of posting a message. When clicking "Post", the cleartext message is retrieved from the input box (1), the plugin retrieves the list of recipients (2), encrypts the message using broadcast encryption (3) and submits the encrypted ciphertext to Facebook (4).

Implementation

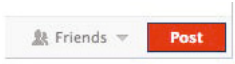
To test the feasibility and practicality of our scheme we implemented a prototype for our system in the context of Facebook. The prototype is an extension for Google Chrome and can be run on both Windows and Mac PCs. The plugin consists of a binary part responsible for performing cryptographic operations and en-/decoding, and a part written in JavaScript responsible for intercepting messages and scanning Facebook pages for encrypted messages.

Interface. Figure 5a shows the main interface of our plugin in Google Chrome. Once the plugin has been initialized and is ready to be used, a user can select whether to encrypt messages or not (for example to allow posting messages in plaintext which are also readable by friends not using the plugin).

(a) prototype plugin



(b) encryption disabled



(c) encryption enabled

Fig. 5. This figure shows screenshots of our prototype. The plugin allows to enable/disable encryption and also dynamically alters the user interface to give visual feedback to the user whether messages will currently be encrypted.

Posts. Figure 2a shows how an encrypted post looks like (i.e., how it would look like to a user which is not using our system).

4 Evaluation

The prototype implementation of our system has all the necessary functionality, but performance is also an important aspect of feasibility and practicality. We therefore ran a series of benchmarks to determine the performance of our system in terms of computing power as well as storage space requirements.

To realize the encryption scheme described in Sec. 2.1, we use the PBC Library[2] for pairing operation and elliptic curve point operations. For the evaluation presented hereafter we used two different curves, A and F.[3] **Curve A** uses a 512-bit base field with an embedding degree of 2. **Curve F** has much smaller elements (base field size 160 bits), and a larger embedding degree of 12.

[2] `http://crypto.stanford.edu/pbc/`

[3] One may use RELIC `http://code.google.com/p/relic-toolkit`, TinyPairing `http://www.cs.cityu.edu.hk/%7Eecc/TinyPairing`, or other bilinear pairing libraries, each may result in different performance figures for various platforms.

Pairings over Curve A are significantly faster than over Curve F (up to 20 times faster[4]), but because of the large number of multiplications and additions in the algorithm the smaller base field size of Curve F (160 bits vs. 512 bits for Curve A) results in better performance for Curve F overall. All experiments were run on a machine with a 2.4 GHz Intel Core 2 Duo processor.

Processing Time

To evaluate the processing time required for the different operations (parameter generation, key extraction, encryption and decryption), we ran benchmarks for each operation and for both curves A and F.

Parameter Generation. Table 1 shows the time required for parameter generation for both curves A and F. The time required is short (less than 6 seconds for both curves) even if $m = 1000$. Furthermore, this only has to be done once when deploying the system and is therefore easily amortized over the normal usage of the system, making it for all practical purposes negligible.

Key Extraction. As shown in Table 1, this is an inexpensive operation requiring only a few milliseconds per user. It can also trivially be parallelized and therefore scales easily to a large number of users if necessary.

Table 1. The CPU time required for different operations

Operation	Curve A	Curve F
Parameter Generation (m=100)	0.62s	0.59s
Parameter Generation (m=1000)	5.8s	4.8s
Key Extraction	5.7ms	1.7ms

Encryption. The processing time required depends on the number of recipients. Fig. 6a shows that encrypting a message destined for 100 recipients requires 1 second for curve F and 1.6 seconds for curve A, which we believe is practical.

Decryption. Figure 6b shows that decrypting a message encrypted for 100 recipients requires less than 0.8 seconds for curve F and approximately 1.1 seconds for curve A. While this seems already reasonable, decrypted messages could also be cached locally on the user's computer, further saving processing time.

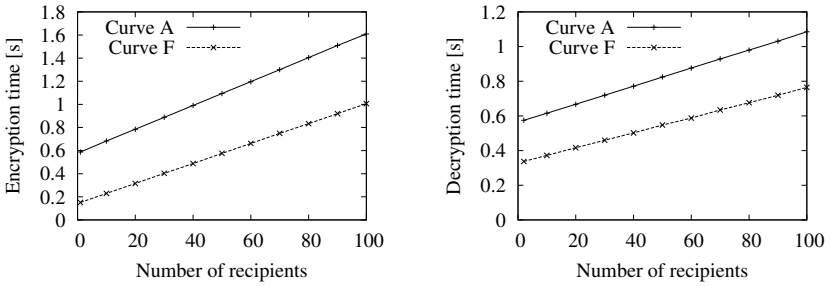
Summary. Considering that the average number of friends of users on Facebook[5] is 190, while the median is 100, we believe that these computation costs are reasonable. Furthermore, adding even a small amount of local caching could reduce the cost even further without compromising privacy.

Size Requirements

There are different objects in our system which need to be stored, e.g., the IBBE master public key, ciphertexts and ciphertext headers.

[4] `http://crypto.stanford.edu/pbc/times.html`

[5] Facebook, Nov 2011, `http://www.facebook.com/notes/facebook-data-team/anatomy-of-facebook/10150388519243859`



(a) Encryption (b) Decryption

Fig. 6. The CPU time required to encrypt/decrypt a message for different numbers of recipients. The CPU time required increases linearly with the number of recipients, and curve F requires less CPU time than curve A.

Public Parameters. The public parameters of our system is the IBBE master public key PK in Sec. 2.1. The size depends on the maximum number of recipients, and Table 2 shows that for $m = 1000$ curve A requires 65KB, while curve F requires less at 41KB. The size of the public parameters is not usually a concern as they can easily be stored within or even hard-coded into the plugin.

Private Parameters. The MSK used by the Key Extraction Server is small (50 bytes for curve F and 100 bytes for curve A).

Private Key. Each user requires a personal decryption key to decrypt messages which is stored locally on a user's computer and which is small in size (Table 2).

Encryption Header. The size of the encryption header is important insofar as each encrypted message requires one, therefore increasing the final size of the ciphertext. In our implementation, the header contains C_1 and C_2. For curve A this amounts to 134 bytes, for curve F to only 66 bytes.

Table 2. Space Requirements: This figure shows the space necessary for different elements in our system for different maximum recipient numbers m

	Curve A		Curve F	
	$m=100$	$m=1000$	$m=100$	$m=1000$
Priv. Param.	93 B	93 B	49 B	49 B
Pub. Param.	6.7 KB	64.7 KB	4.4 KB	41.3 KB
Private Key	68 B	68 B	24 B	24 B
Enc. Header	134 B	134 B	66 B	66 B

Ciphertext. We evaluated the size of the ciphertext compared to the plaintext in terms of display space. Figure 7 shows that while the overhead is more than 150% for very small messages, it quickly drops to 50% overhead for messages of

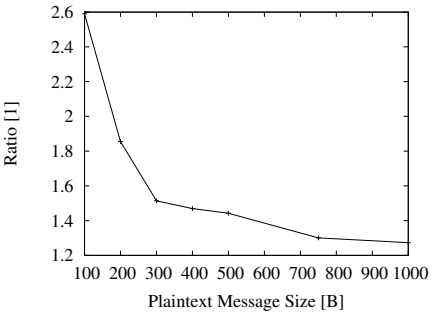


Fig. 7. This figure shows the *approximate* ratio of the *display size* of the ciphertext versus the plaintext (using curve F). Small messages have a larger overhead, but the overhead quickly drops to 30%-50% for plaintexts $\geq$ 300B.

around 300 bytes and then to only 30% for messages of 750 bytes and larger. We argue that for messages of 300 bytes and larger the overheard is acceptable or even small with only 30-50%.

Summary. The size requirements for the different objects in our scheme are in general small enough to be practical, or, for parameters which are larger, can be stored in a way where size is not a concern. Note that one may choose to use other elliptic curve groups with lower bit-strength of the encryption for saving even more space, if the trade-off between space and security is acceptable.

5 Discussions

5.1 Limitations

Dynamic Network. Connections within a social network change, but in our current system, adding somebody as a friend does not give them access to previous posts. A new friend can only access future posts. Similarly, the current prototype only supports forward revocation, which means that a user which is "unfriended", can still decrypt previous posts. Future posts, however, will no longer be accessible to that user. The system could be extended to better support changes in social network, but without active support of the SNP this is likely to require re-encryption of content or some intermediate keys whenever the social network changes. We leave this to future work.

Re-sharing. One issue, which is not limited to our system, is that any user which has legitimate access to a message (i.e., is authorized to decrypt a message) can re-share the message in plaintext, making it potentially available to a much larger audience. This is a fundamental limitation and can be mitigated by only sharing information (even in encrypted form) with other users with whom a trust relationship exists. Similarly, social connections such as "friends of friends" cannot yet be accurately captured. We leave this for future work.

Social Graph. Our system is designed to protect content posted by users, and it can be extended to protect personal information such as the user profile. Hiding the social graph (i.e., who is friends with whom), however, requires a deeper integration with social networking websites than is possible with a simple add-on. Ideally, a privacy-preserving social network website would be designed from the ground up with privacy and security in mind, even hiding the social graph. But due to the network effect the 900 million users on Facebook are unlikely to abandon Facebook for a website with better privacy. Our system can at least protect privacy to a large extent within existing social networking sites.

Opposition of Social Networking Provider. If the business model of an SNP includes monetizing information shared by users, for example for targeted advertising, then the provider may be unwilling to allow people to post encrypted content on the site and for example ban users doing so. This problem is similar to research aimed at protecting privacy in targeted advertising, however, and solutions exist for this problem, e.g. [11,12].

5.2 Future Work

The current prototype demonstrates the basic feasibility and there is still lots of room for improvement.

Subset of Friends. The current prototype by default encrypts messages to all friends of a user. It is straightforward to modify our prototype so that messages can be encrypted to a subset of friends in a more fine-grained way.

Other Content. Currently our prototype only encrypts text content, but it is imaginable to also have protection for other content such as photos.

Other Websites. Our prototype focused on Facebook for reasons of popularity, but the scheme can be generalized to other websites such as Google+ and others.

6 Related Work

Lucas et al. introduced a system called flyByNight [13] which requires an external application server for proxy re-encryption. The server has to be available all the time for users to encrypt and decrypt messages. Our system, on the other hand, does not need any external entity for encryption and decryption, and the key extraction server is used only once by each user when joining the system. Furthermore, the use of JavaScript for client-side encryption in flyByNight severely limits performance, especially for computationally intensive public-key cryptographic operations. Our plugin is written in C and depends on high-performance arithmetic libraries which allows for much better performance. Lastly, their system requires all messaging to be done through a Facebook application, while our system integrates seamlessly with the different parts of Facebook (Wall, etc.).

A system called NOYB was proposed by Guha et al. [14] and it also aims to improve privacy for social network users. It replaces parts of a user's profile with a part of another user such that the social networking site does not even suspect

that information has been faked. The transformation is done using a dictionary and determined using some keys. As a drawback, this scheme requires an out-of-band mechanism to distribute keys and a publicly available dictionary.

Another proposal by Luo et al. [15] is to store fake data on a social networking site, while storing the real data in encrypted form on an external server. When accessing a user's profile, the real data is retrieved from the external server, decrypted and displayed. Their system does not seem to have an "unfriend" mechanism, and it requires the server to be online for any operation, while our scheme is self-contained within the user's browser and only involves one external access when a user joins the system. Their system also requires the distribution of user keys among friends, which is not necessary in our scheme.

Baden et al. propose Persona [16], an online social network which uses attribute-based encryption to allow fine-grained and intricate policies about who can access what. While their system is designed to be a new, independent social network, they also propose integration with Facebook by using Facebook applications, but in a less seamless way than our system. It also requires out-of-band communication for user keys, however, and if Persona is used in conjunction with Facebook it again requires external third-party servers.

Building on the idea of Persona, Jahid et al. [17] present a scheme for access control in social networks using encryption, also providing an efficient mechanism for revocation. They make use of a minimally trusted external proxy and attribute-based encryption allowing fine-grained policies about who can read what information. According to their evaluation performance is decent, with sub-second decryption of messages. Their system, however, does not integrate seamlessly with Facebook and requires a third-party server for all operations.

A different approach for reducing privacy risks in social networks are de-centralized social networks such as diaspora[6] or Appleseed[7], which improve privacy by de-centralizing the storage of shared information. In de-centralized social networks typically nobody has directly access to all information. While this does not yet guarantee perfect privacy, it makes it easier to control information, especially if users can maintain their own node. None of them are in wide-spread use, however, and convincing people to switch from Facebook to any other site is hard because of the network effect.

7 Conclusion

We proposed a broadcast-encryption-based system which allows to share information with friends on social networking sites while at the same time keeping the information private from other users and even from the social networking providers themselves. Apart from a key extraction server which is invoked by a user only during an initial one-time registration, no external servers are needed during normal encryption and decryption operations. We also presented a prototype implementation of our system in the form of a browser plugin and evaluated

[6] `https://joindiaspora.com/`
[7] `http://appleseedproject.org/`

its performance. Results indicated that the system is indeed practical and can protect the privacy of users' information shared on a social networking site.

References

1. Gross, R., Acquisti, A.: Information revelation and privacy in online social networks. In: WPES 2005, pp. 71–80. ACM (2005)
2. Facebook Statistics (November 2011), http://www.facebook.com/press/info.php?statistics
3. Boyd, D.: Facebook's privacy trainwreck. Convergence: The International Journal of Research into New Media Technologies 14(1), 13–20 (2008)
4. Symantec: Facebook Applications Accidentally Leaking Access to Third Parties (2011), http://www.symantec.com/connect/blogs/facebook-applications-accidentally-leaking-access-third-parties
5. MediaPost: Facebook's Data-Leak Woes Worsening (2010), http://www.mediapost.com/publications/article/138132/facebooks-data-leak-woes-worsening.html
6. Digital Trends: Facebook closes loophole that exposes private photos (2011), http://www.digitaltrends.com/social-media/facebook-closes-loophole-that-exposes-private-photos/
7. Acquisti, A., Gross, R.: Imagined Communities: Awareness, Information Sharing, and Privacy on the Facebook. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 36–58. Springer, Heidelberg (2006)
8. Delerablée, C.: Identity-Based Broadcast Encryption with Constant Size Ciphertexts and Private Keys. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 200–215. Springer, Heidelberg (2007)
9. Dhawan, M., Shan, C.-C., Ganapathy, V.: The case for javascript transactions: position paper. In: Proceedings of the 5th ACM SIGPLAN Workshop on Programming Languages and Analysis for Security, PLAS 2010, pp. 6:1–6:7. ACM, New York (2010)
10. Yu, D., Chander, A., Islam, N., Serikov, I.: Javascript instrumentation for browser security. In: POPL 2007, pp. 237–249. ACM, New York (2007)
11. Toubiana, V., Narayanan, A., Boneh, D., Nissenbaum, H., Barocas, S.: Adnostic: Privacy preserving targeted advertising. In: NDSS (2010)
12. Guha, S., Cheng, B., Francis, P.: Privad: Practical Privacy in Online Advertising. In: Proceedings of the 8th Symposium on Networked Systems Design and Implementation (NSDI), Boston, MA (March 2011)
13. Lucas, M.M., Borisov, N.: flyByNight: mitigating the privacy risks of social networking. In: SOUPS (2009)
14. Guha, S., Tang, K., Francis, P.: Noyb: privacy in online social networks. In: Proc. of the First Workshop on Online Social Networks, WOSN 2008, pp. 49–54. ACM (2008)
15. Luo, W., Xie, Q., Hengartner: Facecloak: An architecture for user privacy on social networking sites. In: Computational Science and Engineering, CSE 2009, vol. 3, pp. 26–33 (2009)
16. Baden, R., Bender, A., Spring, N., Bhattacharjee, B., Starin, D.: Persona: an online social network with user-defined privacy. In: Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication, SIGCOMM 2009, pp. 135–146. ACM, New York (2009)
17. Jahid, S., Mittal, P., Borisov, N.: EASiER: Encryption-based Access Control in Social Networks with Efficient Revocation. In: ASIACCS, Hong Kong (March 2011)

Selective and Confidential Message Exchange in Vehicular Ad Hoc Networks

Sushama Karumanchi[1], Anna Squicciarini[1], and Dan Lin[2]

[1] Information Sciences & Technology
Pennsylvania State University
[2] Department of Computer Science
Missouri University of Science & Technology

Abstract. Vehicular Ad-hoc Networks are a promising and increasingly important paradigm. Their applications range from safety enhancement to mobile entertainment services. However, their deployment requires several security issues to be resolved, particularly, since they rely on insecure wireless communication. In this paper, we propose a cryptographic-based access control framework for vehicles to securely exchange messages in a controlled fashion by integrating moving object modeling techniques with cryptographic policies.

1 Introduction

Vehicular Ad-hoc Network (VANET) is an emerging paradigm in which vehicles can communicate with one another and serve as nodes in the network to propagate messages. Such networks comprise of sensors and On-Board Units (OBU) installed in the vehicles as well as Road Side Units (RSU). The data collected from the sensors on the vehicles can be displayed to the driver, sent to the RSU or broadcasted to other vehicles depending on its nature and importance.

Vehicles in VANETs can take advantage of a series of attractive applications, ranging from safety to Internet access and multimedia [5, 7]. Many of these applications cannot rely on simple message broadcasting, but require secure and selective delivery of messages. One such example is the intelligent traffic transportation system which is a popular and widely studied application. The alert system needs to ensure that the alerts do not reach those drivers for whom they are irrelevant, as otherwise they would lead to drivers' distraction [7]. Another emerging application that requires selective message dissemination, is the provision of infotainment services [5] such as music, news, and multimedia via VANETs. Infotainment services are provided only to subscribed vehicles and to vehicles which are in a particular location, and the subscription might further have different authorization levels in its plan. Selective message delivery is also desired for drivers who want to exchange certain messages within a small group of vehicles rather than the entire VANET.

To achieve their full potential, these applications should support an access control mechanism that is able to deliver messages to eligible vehicles that satisfy

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 445–461, 2012.

requirements on both static and dynamic properties, like location. However, access control mechanisms in VANETs are challenging to develop. Vehicles on the road constantly change locations, and periodically change speed and moving directions. It is not trivial to construct access control policies that target message receivers with desired movement patterns, and is also difficult to enforce the access control policies in such a decentralized environment. Some protocols have been proposed to establish cryptographic sessions within groups of vehicles (e.g. [19, 21]), but they provide no guarantee on the honest use of the secret key, and the messages are only exchanged based on identities or static properties.

In this paper, we propose an innovative solution towards supporting access control in VANETs. We leverage moving object modeling techniques and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [2] to capture the dynamic nature of vehicles and strongly enforce access control based on both vehicles' persistent (e.g. year) and dynamic (e.g. location, speed) properties. Our access control scheme provides automated policy generation capabilities for vehicles sending messages to construct policies that cover a reasonable dissemination range in terms of space and time. Projected locations defining the dissemination range are computed using a moving object model. The policies are then encoded using CP-ABE so that only eligible vehicles are able to access the message protected by the policy. Keys and vehicles' attributes are managed by the OBU installed at the vehicle, and the trusted platform module (TPM) [22]. TPMs for automotive systems are not far from reality, and are currently being investigated by both industry [3] and the research community [8, 23].

Our approach has a number of features that make it particularly suitable for the VANET domain. First, it allows vehicles to send out messages in a confidential and secure manner to a selected group of vehicles without the need to know the identities of those vehicles or even be able to link the identity of the message recipient. Second, policies can be specified against dynamic properties of the vehicles, and according to the road topology. Third, the involvement of the RSU can be controlled by the message sender(vehicle). Finally, our framework satisfies a set of critical security properties. Our experiments show efficiency of our proposed approach, for both vehicles and RSUs.

The rest of the paper is organized as follows. Section 2 provides an illustrative scenario. Section 3 presents our model and its goals. Our protocols are presented in Sections 4, 5, and 6. Section 7 analyzes our security properties, while Section 8 reports experimental results. Section 9 reviews related work. Section 10 concludes the paper.

2 Illustrative Scenario

We present the following collaborative driving scenario as an example. Consider vehicles belonging to a goods transportation company, Carly, which assigns tasks to vehicles to transport goods across the country. Carly's vehicles might or might not travel together on the highway. Some of the vehicles may be Carly's temporary subcontractors, whereas others may be Carly's permanent

fleet. Carly's vehicles going toward the same general destination might want to exchange messages regarding the stops in the middle of the journey, number of goods to dispose off at each stop, and so forth. These type of business related messages may be very sensitive and should not be broadcasted to all the vehicles in the proximity. For example, a Carly's vehicle, might want to initiate message exchanges with other vehicles traveling on behalf of Carly, BMW vehicles (for information about shops and BMW retailers), moving in the north direction, and that have not passed exit 50 yet. The initiator vehicle may not want other vehicles, not satisfying the above conditions, to receive its messages. In scenarios similar to the above, where vehicles want to exchange confidential information with selected vehicles, an access control mechanism that takes into account the attribute values or the properties of the recipient vehicles is suitable.

In addition to keeping the messages exchanged confidential from other vehicles not satisfying the policy, other security properties also need to be satisfied. Vehicles exchanging messages may need not know the identities of the vehicles they are communicating with (e.g., Carly's temporary subcontractors). Additionally, a vehicle which satisfied the policy of the initiator vehicle and has exchanged messages with the group might no longer satisfy the same policy once it passes exit 50 and has not taken exit 50. Hence, messages exchanged under this policy should no longer be available to this vehicle. Similarly, a vehicle which joins the Carly travelers' group at a later time, should not receive the messages that were sent before he joined the group. Moreover, if a vehicle does not travel with Carly, it should not be able to access the messages or acquire the authorization to access messages by colluding with a malicious vehicle from Carly.

3 System Model and Approach Overview

3.1 Assumptions and Security Goals

Assumptions. The communicating nodes in VANET are vehicles and roadside units (RSUs). Each vehicle is equipped with an on-board unit (OBU) for the purpose of networking and generating messages, a global positioning system (GPS), a digital map, and a TPM [22]. The TPM is tamper-proof and hence, upon verification of genuineness, it can be trusted to store sensitive key material. The TPM stores data by encrypting with its hierarchy of keys, which are kept secret and are not even revealed to the OBU. Moreover, each vehicle is uniquely qualified by a set of identifying attributes which are classified as persistent and dynamic attributes. *Persistent attribute* values remain constant throughout the lifetime of a vehicle or for an extended period of time. The color, year, and the brand, are examples of persistent attributes. *Dynamic attribute* values change frequently, such as current location of the vehicle, speed and direction.

RSUs are distributed across the roads in a uniform fashion to provide infrastructure support for network setup and communication in VANETs. In line with the assumptions usually accepted for VANETs [14, 16, 20], we assume RSUs are trusted and can be synchronized with other RSUs. Each RSU has a tamper-proof

device for storing secure information. RSUs may also retrieve vehicle information from resources such as the DMV (Department of Motor Vehicle) database.

Vehicles communicate with one another and with RSUs using data link technology (e.g., ASTM E2213-03 [1]), within a range of 10s minimum travel time (the minimum range is 110 m and maximum is 300 m) [15]. Each RSU has a communication range of 300 meters and uses scheduling algorithms to ensure an acceptable message throughput with vehicles [25].

Our threat model is simple: An attacker may try to obtain a policy protected message or collude with other vehicles to obtain the decryption key for an encrypted message.

Security Goals. Bearing in mind the example discussed in Section 2 and other relevant applications where selective delivery of messages is desired, we identify the following set of requirements for selective disclosure of messages across vehicles. These are related to feasibility and to security. With respect to feasibility, we identify the following requirements: (a) Messages should be delivered selectively to vehicles according to criteria that capture the dynamic and structured nature of the VANETs. These include road topology, vehicles' locations and velocities. (b) The process should not introduce significant communication overhead. Finally, (c) Some basic access control capabilities should be available even in case of temporary unavailability of RSUs. Concerning security, we aim to design a system meeting the following properties:

1. *Non-linkability of Messages*: Let m_{p1}, m_{p2} be two messages produced by a vehicle under two distinct policies $p_1,\ p_2$ over a protocol-selectable period of time τ sent at time τ_1 and τ_2 respectively, such that $\tau_2 > \tau_1 + \tau$. m_{p1}, m_{p2} appear unrelated from a recipient vehicle's perspective.
2. *Forward Confidentiality*: A vehicle should not be able to access any messages protected by a policy that it does not satisfy. If the vehicle satisfied a policy at time τ_0 but no longer at time τ_1, from τ_1 onwards all the messages under the same policy should not be accessible by the vehicle.
3. *Backward Confidentiality*: Given a policy-protected message m sent at time τ_1, if the vehicle satisfies the policy at time τ_2 but not at τ_1, where $\tau_2 > \tau_1$, the vehicle should not be able to access the message sent at τ_1.
4. Protection against *Collusion Attack*: Any unauthorized vehicle should not be able to access policy protected messages by sharing the secret key or by pooling with other vehicles the information required to satisfy the policies.

We do not list anonymity and authentication as our security goals since we focus on secure message delivery: as shown in the next sections, our protocol relies on attributes of the vehicles for secure exchange, rather than on their identities. In our framework, a vehicle neither needs to reveal its long-term identity nor needs to know the identity of the recipient when sending a message. As for authentication, we require the vehicles to undergo authentication with the RSU during the registration phase. If needed by the application, our protocol can be preceded by other existing authentication systems.

3.2 Overview of VANET Access Control Schema

Our proposed VANET access control framework consists of three main phases: (1) registration; (2) automatic policy generation; and (3) secure message exchange.

At the registration phase, vehicles need to register at a nearby RSU and get their OBUs attested by the RSU, while the RSU records the vehicle's static and dynamic attributes. After registration, the sender vehicle (message sender) specifies the access conditions that recipient vehicles should satisfy, in order to deliver messages in a controlled manner. These access conditions are then represented using the attribute-based access control (ABAC) paradigm [24], and this process is conducted automatically by our developed policy manager software, installed in the OBU of the vehicle. We model the policies as Boolean expressions against vehicles' attributes. An example policy from the scenario in section 2 looks as follows: $\{company = Carly$ AND $brand = BMW$ AND $direction = North$ AND $location <= Exit50\}$.

To assist the sender in efficiently specifying policies for vehicles within its communication range for a non-null time period, our solution includes an approach to calculate the range of dynamic attribute values representing the surrounding vehicles' projected trajectories. The values are computed by the policy manager installed in the OBU. The vehicle's relative position on a road is modeled as a linear function: $l(t) = l_u + \delta \cdot v \cdot (t - t_u)$, where l_u is the vehicle's distance to the starting point of the road at time t_u, δ is vehicle's moving direction along the road, and v is the vehicle's speed. $l(t)$ predicts the vehicle's position at a near future timestamp t ($t \geq t_u$) assuming that vehicles move at current speed and direction.

Once a policy is specified, the message is encrypted according to its policy. We employ the CP-ABE encryption scheme [2], since it has a number of interesting features that suit well our access control problem. In particular, CP-ABE supports the notion of attribute-based policies as a criteria for encryption, affords strong security and privacy guarantees, and does not require prior identity information of the recipient.[1] According to the CP-ABE scheme, the OBU uses the policy in conjunction with a protocol-specific public key to encrypt the message. The secret key required to access a policy protected message is stored in the TPM of the vehicle, and distributed in two ways: (a) The sender vehicle requests the RSU to compute and distribute secret keys to the eligible vehicles; (b) The vehicles on road, request the RSU for a secret key in certain time intervals or when their attribute values change.

In both cases, the TPM releases the locally stored secret key to the OBU, so that the OBU can attempt to decrypt the message. The key is released only when the vehicle satisfies the attribute values based on which the message is created.

When vehicles move across RSUs, vehicles' information is also passed along to the next RSUs. Specifically, the RSU computes the estimated leaving time of

[1] The technical report at `http://www.personal.psu.edu/sik5273/NSS2012/TechReport.pdf` provides detailed information on the background of CP-ABE and TPM, as well as detailed protocols.

each vehicle using the vehicle's current movement function. When the leaving time is approaching, if the RSU has not received any new update from the vehicle, the RSU will assume that the vehicle is still traveling at its previously reported speed and direction, and hence concludes that it will soon be out of the RSU's communication range. The RSU will forward the vehicle's information to the next RSU in the vehicle's direction.

4 Vehicle Attestation and Registration

The RSUs store the master key, MK, an ABE public key, PK_{ABE}, both generated according to the CP-ABE scheme, a public key, PK_{RSU} and secret key, SK_{RSU}. The master key is known only to the RSUs and is shared across all and only RSUs in the VANET, while the PK_{ABE} is public. Any vehicle which wishes to use the RSU infrastructure to exchange messages, undergoes an attestation phase followed by a registration phase at a nearby RSU, at the start of a trip. The attestation process aims at authenticating a vehicle by verifying whether the vehicle has a genuine TPM and whether the OBU has a valid and uncorrupted software installed. The registration phase aims at obtaining the attribute values of the attested vehicles. The protocol for the OBU attestation consists of verifying the TPM's PCRs and measurement lists, and is carried out according to the TCG guidelines. The TPM is identified to be genuine by the RSU by verifying the TPM certificate.

Upon having verified the validity of the software of the OBU, the vehicle registers its attribute values with the local RSU. The vehicle begins by broadcasting a registration request by specifying the vehicle's license plate number, LN, signed using the TPM's private attestation identity key, SK_{AIK} and encrypted using the public key of the RSU. The RSU, upon receiving the request, retrieves all the persistent attributes (e.g., car brand) of the vehicle from the vehicle databases such as DMV, and requests the vehicle for its dynamic attribute values, that is, its current location, speed and direction values. The vehicle replies by sending a message whose main components include the current attribute values for location (l), velocity (v) and direction (δ). The RSU verifies the correctness of the received attribute values. For example, it employs location proof methodologies for location-related attributes [18] and its radars for detecting speed and calculating direction. The registration ends with the RSU sending a confirmation message to the vehicle, along with a set of expected values for the attributes with a short lifetime expectancy. A vehicle registers at the start of its trip, and gets re-registered and re-attested when it loses its private key received from the RSU, that is, in cases when the OBU gets rebooted (such a case is explained in section 7.2).

Over time, to inform about movement, each vehicle sends beacon signals to the RSU periodically. Verification of attributes is executed in two cases. One case is that the vehicles send new updates to the RSU when vehicles' speed or direction is changed drastically (e.g. the vehicle stops suddenly or exits a highway to enter a slow traffic urban area). The other case is that a RSU-established time interval is elapsed and the vehicle needs to send a status update to the RSU.

5 Access Control Policies and Message Generation

To establish a lasting communication link, messages should be exchanged not only according to individual vehicles' attributes, but also to a combination of properties associated to vehicles' movements, such as velocity and location, and other static properties. Accounting for this dynamic information allows the vehicles to communicate with vehicles who are in the proximity and whose trajectory is similar. Speed and location are however intrinsically dynamic, and it is not possible to settle down on a single static value for moving vehicles. To overcome this issue, we assume that dynamic attributes are discretized into ranges (e.g. 60-65mph, or location detected every 300 meters) and that, as they are received by the RSU or the TPM, they are always considered within an error range ϵ specific for the attribute type.

5.1 Policy Generation

We build on the above assumptions to propose an approach to compose policies taking into account the vehicles' movement. The key idea is to extract common movement patterns from the desired recipient vehicles, and then specify such projected patterns using attribute-based access control (ABAC) policies [24]. We consider two main types of applications to classify movement patterns: (i) traffic or safety message dissemination (e.g., car accident, traffic congestion, road flooding); (ii) entertainment applications (e.g., vehicles looking for traveling partners). These applications, which are most interesting for our selective message delivery approach, will be supported by automatically generated policies, so as to reduce the burden of policy configuration for sender vehicles. More importantly, policies generated by our approach provide some guarantees of applicability, in that they account for the mobility of the vehicles and identify the vehicles that are on the same trajectory of the sender, and at communication distance for a time interval long enough to establish the communication. Operationally, users will need to indicate the purpose of the message to be sent out and possible requirements against persistent attributes, and a policy for the application will be generated.

Consider messages related to intelligent alerts. These type of messages are typically relevant for vehicles within certain distance and traveling towards the scene being alerted (e.g. a car accident, a traffic jam). The policy for these types of applications is thus defined by three factors: the event location (l_e), the distance threshold (λ) and the moving direction (δ_x, δ_y) on x- and y-axes of the map. The distance threshold λ is estimated as the multiplication of the speed limit and the historical average time of the traffic being clear, to identify the vehicles within this range which may be affected by this event. The moving direction is either 1 or -1, indicating the direction towards the event location. Finally, the following ABAC policy P_{trf} will be generated:

$$P_{trf}\colon (Dist(l_e, l_r) < \lambda) \wedge (\tfrac{v_x}{\delta_x} > 0) \wedge (\tfrac{v_y}{\delta_y} > 0)$$

where $Dist$ is the function to compute the distance between the current location of a recipient vehicle (l_r) and the event location (l_e), and (v_x, v_y) is the velocity of the recipient vehicle. Policy P_{trf} says that vehicles which are less than λ miles

away from and moving towards the event location will be able to receive the message related to this traffic event. The policy is eventually encoded in terms of differences between relative locations so that the distance conditions can be checked against verified attributes.

As another example of message type, let us consider messages aiming at collaborative driving. For a vehicle that initiates a connection, the system will look for vehicles with similar movement patterns as the vehicle so that they may travel together for a relatively long time period. Other criteria against persistent attributes, such as car make, year, final destination, etc, may be considered by the sender to select possible desired recipients. Yet, regardless of the criteria defined over persistent properties, the key issue is to quantify the movement similarity. We adopt the following criteria: the maximum distance between the sender vehicle and the recipient vehicle should not exceed the communication range r within certain time period Δ_τ, where Δ_τ is determined according to the maximum time interval between two consecutive updates of their attribute values in the history. Recall that under the linear function modeling, vehicles only need to send an update to the RSU when their speed or moving direction has been changed dramatically. Vehicles' locations, before their next updates, can be predicted using their latest location and velocity information. Accordingly, we obtain the following function of distance (square of the distance) between the sender vehicle and the recipient vehicle at a future timestamp τ:

$$D(\tau) = [(l_s + \delta_s \cdot v_s \cdot \tau) - (l_r + \delta_r \cdot v_r \cdot \tau)]^2 \tag{1}$$

In Equation 1, l_s and $\delta_s \cdot v_s$ denote the current location and velocity of the sender vehicle, and l_r and $\delta_r \cdot v_r$ denote the current location and velocity of a recipient vehicle. This distance function projects vehicles' future positions on the same road. We argue that a vehicle is very likely to send out another update due to the change of trajectory when it enters a new road or passes an intersection. Therefore, the estimation using the above linear functions is expected to be close to the true distance between two vehicles before they send new updates.

$$D(\tau) = (\delta_s \cdot v_s - \delta_r \cdot v_r)^2\tau^2 + 2(l_s - l_r)(\delta_s \cdot v_s - \delta_r \cdot v_r)\tau + (l_s - l_r)^2 \tag{2}$$

To find the maximum distance within the time interval $[\tau_c, \tau_c+\Delta_\tau]$, we reorganize Equation 1 to be the function of τ as follows. If the coefficient of τ^2 is 0, i.e., $(\delta_s \cdot v_s - \delta_r \cdot v_r)^2 = 0$, the sender vehicle and the recipient vehicle have the same velocity. Thus, their distance is a constant $\sqrt{(l_s - l_r)^2}$. In other cases, the coefficient of τ^2 is positive, and Equation 2 is a second degree polynomial function of τ. Its graph is a parabola that opens upward. The maximum distance is reached either at τ_c or $\tau_c+\Delta_\tau$. Therefore, the policy conditions over dynamic attributes are configured as follows. P_{soc}: $(D(\tau) < r^2) \wedge (D(\tau + \Delta_\tau) < r^2)$.

5.2 Message Creation

Once the access policy P is computed, the OBU of the sender vehicle, constructs an access structure, $\mathcal{T}_p$, which provides a normalized representation of the policy

as a conjunction and/or disjunction of Boolean formula on a set of attributes. The OBU then runs the $Encrypt(PK_{ABE}, \mathcal{T}_p, m)$ primitive of the CP-ABE family, to generate the ciphertext message CT from plain text m under the access structure $\mathcal{T}_p$. The sender vehicle also includes the message creation time t_m and the message expiration time τ_e to be sent along with the message. The validity interval is either inserted as part of the policy structure, or it is appended to the message, in which case is signed by the RSU to ensure the integrity of the validity time. For the latter case, an example of message m of content C targeting BMW vehicles driving NORTH is: $\{C, Sig_{SK_{RSU}}(t_m$=1:14 pm, τ_e= 1:24pm, $n_{RSU})\}$. n_{RSU} is a nonce created by the RSU to prevent replay attacks, and it is sent with the secret key generated according to the message request. The sender broadcasts CT: $\{C, Sig_{SK_{RSU}}(t_m$=1:14 pm, τ_e=1:24pm, $n_{RSU})\}_{\{\mathcal{T}_p, PK_{ABE}\}}$, where $\mathcal{T}_p$ is (Make = BMW AND Dir= NORTH).

6 Secure Message Exchange Protocols

Secure message exchange entails two key steps: (1) Secret key delivery to the potential recipients (2) Secure decryption of the message at the vehicle.

6.1 Secret Key Delivery Strategies

We support two delivery strategies. The first one (Secret Key Generation on Message Delivery) allows generation of keys on message delivery. The second one, referred to as Frequent Secret Key Generation, supports keys generation only upon explicit request of a vehicle, therefore limiting the overhead at the RSU. As a result, the latter strategy suits well cases where RSUs are not consistently available, or they are overloaded by other management tasks, and when the secret keys are computed only over persistent attribute values. Figure 1 depicts the information flow of the two approaches.

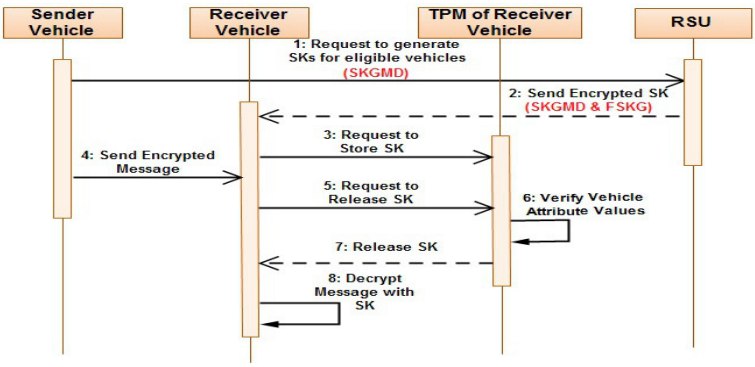


Fig. 1. Key steps of the message delivery protocols

Secret Key Generation on Message Delivery (SKGMD): Under this strategy, the sender vehicle requests the RSU to generate a secret key for the eligible vehicles by sending the desired attribute values for each attribute, called *ranges* (message 1 in Figure 1). The request is securely encrypted using the public key of its nearby RSU, and it includes a nonce n_{V_s} to prevent replay attacks and its license number LN. The nearby RSU upon receiving the message completes the key establishment protocol. First, the RSU identifies the vehicles that can potentially match the given *ranges*. The actual identification process of candidate vehicles is based on currently available information of the vehicles in its network and their projected dynamic attribute values. Once potential recipients are identified, the RSU deploys the CP-ABE cryptographic primitive $KeyGen(MK, S)$, and generates the message decryption key SK_S for each recipient. S is the set of values of attributes of the recipient vehicle. SK_S is unique to each vehicle based on its attribute values.

Frequent Secret Key Generation (FSKG): In this strategy, the RSU delivers SKs to all the registered vehicles in its proximity in certain time intervals (message 2 of Figure 1) or when vehicles request for an updated key. If a vehicle has no valid SK_S to decrypt a message, it may request one from the nearest RSU by sending a signed old key to the RSU for verification of its genuineness (an associated attack is described in Sect. 7.2). The RSU verifies the old key of the vehicle, and issues a new secret key to the vehicle's TPM.

6.2 Message Decryption

To access a message, the vehicle needs to have the proper decryption key, delivered according to either the FKSG or the SKGMD approach. Regardless of the strategy, the key is always obtained from the RSU, which sends message $\{M_p\}_{PK_{AIK}}$ to the TPM of the recipient vehicles. $M_p = \{Sig_{SK_{RSU}}(SK_S||S)\}$, where SK_S is the secret key on attribute values S. M_p is encrypted using the public AIK, PK_{AIK} of the TPM of the recipient vehicle. This ensures that only the TPM component of the vehicle can access the key. The OBU of the vehicle requests the TPM to store SK_S and S (message 3 of Figure 1). The TPM stores SK_S and S within its protected storage. SK_S's lifespan is inherently determined by the expected lifetime of the attributes used to generate it. Upon receiving the message, the following steps are carried out:

- The OBU of the recipient vehicle requests its TPM to release the secret key SK_S to decrypt CT (message 5 of Figure 1).
- The TPM requests the OBU to provide to it the current attribute values of the vehicle, S'. The TPM now compares the current values with S, and if $S = S'$ (message 6), releases SK to the OBU (message 7).
- The OBU decrypts the message using SK (message 8). (A related attack is discussed in Sect. 7.2).

7 Security Properties

We now analyze the security properties achieved by our scheme, followed by a discussion of relevant attacks to our scheme. The discussion of classic attacks is omitted due to lack of space. For example, replay attacks are prevented by means of standard measures such as nonces and timestamps. No policy is ever disclosed in clear to any vehicle, and no encryption is carried out outside the direct control of the TPM of each vehicle.

7.1 Properties

Forward confidentiality is achieved by ensuring that a vehicle cannot access a given message once its attributes do not satisfy the policy specified by the sender vehicle. We require every vehicle to update each of its dynamic attribute values (e.g. l, v), to its TPM in frequent time intervals specified by the RSU. A comparison application is executed by the Trusted Computed Group Software Stack (TSS- part of TPM). The application compares the dynamic attribute values sent by the OBU with its stored dynamic attribute values in S received from RSU. If the TSS finds that the stored values are equal to those provided by the OBU, the key constructed using attribute values S is still valid. Otherwise, the TSS considers the corresponding key(s) expired, and notifies the TPM to delete the old key(s) and requests a new key on the new attributes from the nearby RSU. Consequently, the vehicle will not be able to open the message encrypted using the old key(s), and thus, forward confidentiality is achieved.

In order to satisfy *backward confidentiality*, a vehicle which qualifies to access a message according to a policy p at time τ_o, should not be able to access the messages that are exchanged under the same policy p before τ_o. The vehicle might however store the encrypted message previously received, and try to decrypt it using SK_S upon obtaining it (assume this is obtained at a time $\tau_1 > \tau_o$). This breach is avoided by the controls performed by the algorithms in the TSS. The TPM obtains expiration time τ_e and creation time t_m of the message along with the encrypted message CT. Using these time stamps, the TSS checks if the message can be opened. If allowed, the TPM releases the SK_S for decryption. Otherwise, the TPM does not release SK_S, therefore guaranteeing backward confidentiality.

Anonymity is also guaranteed. No ID or persistent attribute is ever needed to establish a policy-protected communication. The keys for message exchange are not tied to one another and do not require previous vehicle-to-vehicle agreements, avoiding any linkage.

7.2 Attacks

In this section, we discuss a slew of attacks that may be launched against our protocols, and present how our framework prevents these attacks.

Collusion Attack: Malicious vehicles in the network can instigate a *collusion* attack. This attack is achieved by a vehicle that does not satisfy the policy of a

group, obtaining the authorization to access the private key that satisfies the policy from a malicious vehicle satisfying the policy. In our protocols,since the private key is not disclosed to the vehicle but securely stored in its TPM, the vehicle cannot share the private key with any other vehicle. Further, collusion attacks that are carried out by attempting to compose a key using attribute values belonging to different vehicles will fail, due to inherent properties of CP-ABE and our verification protocols. First, the CP-ABE schema prevents attribute sharing by randomizing users' private keys and ciphertexts such that they cannot be combined. Second, the verification protocols carried out by the RSU when new attribute values are proposed ensure that vehicles' attributes are in fact generated by a single vehicle.

Masquerade Attack: A vehicle may launch a masquerade attack, in which it pretends to possess a specific set of dynamic attributes in order to access one or more policy-protected messages. An attacker may try different attribute values to gain the access. Our framework prevents such attack by having the RSU verify the validity of the attribute values sent by a vehicle. The attacker will only receive keys generated based on its true attribute values, and will not be able to obtain keys on any fake attributes. Further, by construction, the message will either be successfully decrypted or will not be. No other feedback will be provided, avoiding policy guessing. Alternatively, the OBU might provide false attribute values to its TPM, therefore masquerading its real identity. This type of attack may succeed only if the attacker can tamper the original software in charge of sending the values. For example, the attacker tries to add a module that generates false attribute values. For the changes to be effective to the OBU, the OBU needs to be rebooted. Our framework requires that a vehicle with a rebooted OBU to delete the secret key and get re-attested (Sect 4), therefore preventing these attacks.

Compromised Vehicle Attack: A driver of the vehicle might compromise the OBU and reboot it, and then request the RSU for the renewal of a key pretending to be a genuine vehicle without getting re-attested. After the registration phase, the RSU cannot differentiate a key request from a genuine attested vehicle and a compromised vehicle which was previously attested. For this purpose, we require each vehicle requesting a renewal of the key to provide its old key to the RSU. Since a compromised vehicle reboots its OBU, the TPM deletes the old key. Hence, it cannot provide the old key to obtain a new key.

Memory Access Attack: An attacker might access the memory of the OBU when the secret key is released into the memory and when the message is being decrypted, to obtain the secret key. To achieve this, the attacker needs to modify the kernel code of the OBU. For this, the attacker needs to reboot the OBU. When the OBU reboots, the TPM of the vehicle deletes the secret key stored within it. The attacker after rebooting the machine, will lose the secret key stored within the TPM. If the attacker attempts to request a new key from the RSU, the attacker has to provide the current key that it does not have. In any

case, the attacker has to re-register and get re-attested with the RSU for further message exchange in VANETs.

Cold Boot Attack: The cold boot attack exploits the fact that a Dynamic RAM (DRAM) possesses a remanence characteristic wherein even after the system is shut down, the DRAM still retains the contents of the memory for some small amount of time. In our scenario, a fraud driver could shutdown the OBU in order to retrieve the decryption keys in the OBU's memory when released during message decryption. To mitigate this attack, we require the OBU to clear the memory on frequent intervals. Further, risks of successful cold boot attacks can be mitigated also by clearing RAM at startup using Power-On Self-Test before the operating system loads, and by letting the operating system identify those memory locations that decay quickly and to store the keys in those positions [10].

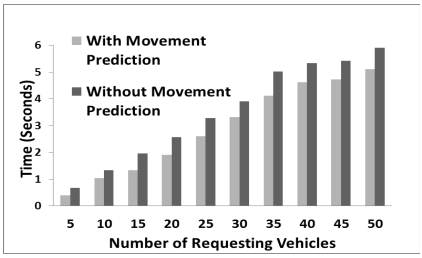
8 Performance Study

The experiments were conducted on an Intel core i7 2630QM CPU @2.00 GHz, 8GB RAM, Ubuntu 5.6 OS. We evaluate the efficiency of the secret key establishment protocol, hence the overhead at the RSU. For computing communication/network delays, we use the Network Simulator-2 package. For all the experiments, we consider up to 50 vehicles, determined as follows. The best transmission range of the RSU is 300 meters [15]. The average length of each vehicle is 4 meters and the vehicles maintain safe distance, and hence one RSU could cover up to 50 vehicles. We let each vehicle move with a speed of about 60 mph. According to the transmission range of RSU, a vehicle moving at 60 mph stays within the range of one RSU for around 11 seconds. We also consider the overhead introduced by the TPM. One TPM encryption using an RSA algorithm takes around 60 ms and one decryption takes around 500 ms [9].

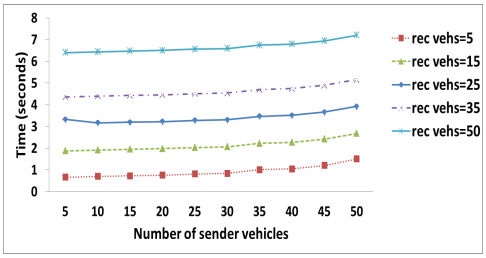
Simultaneous Requests and Movement Prediction. The first set of experiments measures the time to compute and distribute the secret key (SK) by the RSU under the *FSKG* strategy where a group of vehicles request the RSU for SKs[2]. We record the total time for the following three steps: (1) Each vehicle in the communication range of a certain RSU requests the RSU for SK by sending its attribute values; (2) The RSU computes SK for each requesting vehicle based on the vehicle's attribute values, that is, we measure the computation time of the CP-ABE scheme to generate SK; (3)The RSU then encrypts SK with the AIK of the TPM of each vehicle and broadcasts the encrypted SK. In the experiment, we vary the number of requesting vehicles simultaneously requesting SKs from 5 to 50, and set the number of attributes per vehicle to 5.

Figure 2(a) shows the time taken by the two approaches, that is, with and without using the projected trajectory information. First, we observe that the computation and communication time increases almost linearly with the number of requesting vehicles requesting SKs at the same time. In the worst case,

[2] Recall that dynamic properties, such as location, are updated periodically. If updated every 400 meters, a new corresponding key is requested at most once per RSU.



(a) Simultaneous SK requests



(b) Sender and recipients ratio by group

Fig. 2. Experimental Results

when the RSU's communication range is crowded with 50 vehicles, the time for key generation and distribution for all 50 vehicles is less than 6 seconds and is around 2.8 seconds for 25 vehicles. Hence, each vehicle can comfortably request and receive an SK from a single RSU. Second, the test demonstrates that including conditions against the projected trajectory in the policy always results in a performance improvement, as compared to not using any such conditions. The use of the movement prediction in fact significantly reduces the amount of location updates sent from vehicles. This in turn reduces the chance of message jam and also lessens the workload at the RSU since the RSU performs decryption for fewer number of updates.

Policy Complexity. In the second experiment (not shown in the figure), we vary the number of attributes, ranging from 1 to 10, included in a policy and measure the time taken to compute and distribute the secret keys, in groups of key requesting vehicles of size 10, 30 and 50. The time increases proportionally with the number of attributes. This is because, the more the attributes, the more the computation needed to generate the secret key. We found that even when 30 vehicles request keys at the same time, all of them can receive their keys computed on 10 attributes within about 5 seconds.

Sender-Receiver Ratio. We evaluate the *SKGMD* strategy (Figure 2(b)). We vary both the number of sender vehicles and eligible recipient vehicles from 5 to 50, and set the attributes in each policy to 5. We assume all the sender vehicles send the request to the RSU at the same time, and measure the total time taken for the recipients to obtain the keys. The total time increases slightly with the number of recipient vehicles while it is not affected much by the increase of the sender vehicles. Under the *SKGMD* strategy, the RSU just needs to compute one secret key for each eligible recipient vehicle based on the vehicle's attributes. The same secret key can be used by the vehicle to decrypt all the messages protected by the policies that the vehicle satisfies. Irrespective of the number of senders (i.e., policies), the time for key computation is the same as long as the number of eligible vehicles remains constant. The slight increase in time with the increase of number of sender vehicles is mainly due to the time taken to evaluate the increased number of policies. This experiment confirms that secret keys can be reused when vehicles' attributes do not change, limiting reliance on the RSU.

9 Related Work

While attribute-based access control is a well-studied topic (e.g. [17]), access control in VANET is relatively unexplored. Hong and colleagues propose Situation Aware Trust (SAT)to provide adaptive and proactive security in mobile scenarios like VANET [12]. Attributes in SAT identify a group of entities, the type of application, or the property of events. Subsequently, Chen et al. propose message dissemination algorithms based on vehicles' attributes [4]. We differ from previous work in scope, techniques and purpose, focusing on access control, selective message exchange, vehicles's movements rather than on group policies, and our approach assures stronger message confidentiality. A similar approach is taken by Huang et al. [13], where CP-ABE is used for policy enforcement in VANET. However, we differ from the work in many ways. Huang et al. do not address the problem of private key collusion attack which is a critical security concern. The whole scheme fails if collusion attacks are not prevented. Also, they do not deal with the problem of key expiration which is a unique problem in dynamic environments such as VANETs. Finally, we consider the vehicle trajectories to suggest policies for the user. The benefits of TPM in VANETs have been acknowledged [8,23]. In [8], Guette and Heen proposed a TPM-based security architecture, which relies on a cache of securely stored keys, managed by the TPM. The primary goal is to preserve anonymity while facilitating cooperative driving.In the current work, we leverage TPM's security properties to ensure secret key storage and management. To limit the overhead and the reliance of possibly unsafe caches and updates of TPMs, we do not allow caching keys.

Message authentication in VANETs has been long explored, and [6,11,20] are some examples. Works so far address authentication issues while guaranteeing properties such as traceability and privacy, but do not tackle the issue of selective message exchange.

10 Conclusion

We proposed a novel framework for attribute-based access control in VANETs. Our framework combines the advantages of the CP-ABE scheme, the TPM technology and moving object techniques to offer a convenient way for vehicles to exchange messages in a controlled manner. Next, we plan to refine the verification mechanisms for vehicles' attributes, study how to reduce reliance on the RSU, and relax the assumptions of trustworthiness.

References

1. Standard specification for telecommunications and information exchange between roadside and vehicle systems, `http://www.astm.org/Standards/E2213.htm`
2. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy, SP 2007, pp. 321–334. IEEE Computer Society, Washington, DC (2007)

3. Channel, A.: Toyota motor corporation joins trusted computing group, tcg (2012), http://www.theautochannel.com/news/2012/03/15/029016-toyota-motor-corporation-joins-trusted-computing-group-tcg.html
4. Chen, N., Gerla, M., Huang, D., Hong, X.: Secure, selective group broadcast in vehicular networks using dynamic attribute based encryption. In: The 9th IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), pp. 1–8 (June 2010)
5. Cheng, H.T., Shan, H., Zhuang, W.: Infotainment and road safety service support in vehicular networking: From a communication perspective. Mechanical Systems and Signal Processing 25(6), 2020–2038 (2011)
6. Chuang, M.-C., Lee, J.-F.: Ppas: A privacy preservation authentication scheme for vehicle-to-infrastructure communication networks. In: Consumer Electronics, Communications and Networks Conference (CECNet), pp. 1509–1512 (2011)
7. Dobre, C., Fratila, C., Iftode, L.: An approach to evaluating usability of vanet applications. In: 2011 7th International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 801–807 (July 2011)
8. Guette, G., Heen, O.: A TPM-based architecture for improved security and anonymity in vehicular ad hoc networks. In: 2009 IEEE Vehicular Networking Conference (VNC), pp. 1–7 (October 2009)
9. Gunupudi, V., Tate, S.: Timing-accurate TPM simulation for what-if explorations in Trusted Computing. In: International Symposium on Performance Evaluation of Computer and Telecommunication Systems, pp. 171–178 (2010)
10. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: cold-boot attacks on encryption keys. Commun. ACM 52, 91–98 (2009)
11. Hao, Y., Chengcheng, Y., Zhou, C., Song, W.: A distributed key management framework with cooperative message authentication in vanets. IEEE Journal on Selected Areas in Communications 29(3), 616–629 (2011)
12. Hong, X., Huang, D., Gerla, M., Cao, Z.: SAT: situation-aware trust architecture for vehicular networks. In: 3rd International Workshop on Mobility in the Evolving Internet Architecture, MobiArch 2008, pp. 31–36. ACM (2008)
13. Huang, D., Verma, M.: ASPE: attribute-based secure policy enforcement in vehicular ad hoc networks. Ad Hoc Networks 7(8), 1526–1535 (2009)
14. Jung, C.D., Sur, C., Park, Y., Rhee, K.-H.: A robust conditional privacy-preserving authentication protocol in Vanet. Social Informatics and Telecommunications Engineering 17, 35–45 (2009)
15. Lee, J.: Design of a network coverage analyzer for roadside-to-vehicle telematics networks. In: International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp. 201–205 (2008)
16. Lu, R., Lin, X., Zhu, H., Ho, P.-H., Shen, X.: ECPP:efficient conditional privacy preservation protocol for secure vehicular communications. In: IEEE Conference on Computer Communications, pp. 1229–1237 (2008)
17. Mohan, A., Blough, D.M.: An attribute-based authorization policy framework with dynamic conflict resolution. In: Proceedings of the 9th Symposium on Identity and Trust on the Internet, IDTRUST 2010, pp. 37–50. ACM, New York (2010)
18. Saroiu, S., Wolman, A.: Enabling new mobile applications with location proofs. In: 10th Workshop on Mobile Computing Systems and Applications (2009)
19. Studer, A., Bai, F., Bellur, B., Perrig, A.: Flexible, extensible,and efficient vanet authentication. Journal of Communications and Networks 11(6), 574–588 (2009)

20. Studer, A., Shi, E., Bai, F., Perrig, A.: Tacking together efficient authentication, revocation, and privacy in VANETs. In: 6th Annual IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pp. 484–492 (2009)
21. Sun, J., Zhang, C., Zhang, Y., Fang, Y.: An identity-based security system for user privacy in vehicular ad hoc networks. IEEE Transactions on Parallel and Distributed Systems 21(9), 1227–1239 (2010)
22. T.M. Trusted Computing Group Published. Part 1, design principles, specification version 1.2, level 2, revision 103 (July 2007)
23. Wagan, A., Mughal, B., Hasbullah, H.: VANET Security Framework for Trusted Grouping Using TPM Hardware. Communication Software and Networks, 309–312 (February 2010)
24. Wang, L., Wijesekera, D., Jajodia, S.: A logic-based framework for attribute based access control. In: 2004 ACM Workshop on Formal Methods in Security Engineering, FMSE 2004, pp. 45–55. ACM (2004)
25. Zhang, Y., Zhao, J., Cao, G.: On scheduling vehicle-roadside data access. In: 4th ACM International Workshop on Vehicular Ad Hoc Networks, VANET 2007, pp. 9–18. ACM (2007)

Cryptanalysis of Two Dynamic ID-Based Remote User Authentication Schemes for Multi-server Architecture

Ding Wang[1,2,⋆], Chun-guang Ma[1], De-li Gu[1], and Zhen-shan Cui[1]

[1] College of Computer Science and Technology, Harbin Engineering University, Harbin City 150001, China
[2] Automobile Management Institute of PLA, Bengbu City 233011, China
wangdingg@mail.nankai.edu.cn, machunguang@hrbeu.edu.cn

Abstract. In NSS'10, Shao and Chin pointed out that Hsiang and Shih's dynamic ID-based remote user authentication scheme for multi-server environment has several security flaws and further proposed an improved version which is claimed to be efficient and secure. In this study, however, we will demonstrate that Shao-Chin's scheme still cannot achieve the claimed security goals, and we report its following flaws: (1) It cannot withstand offline password guessing attack under their non-tamper resistance assumption of the smart card; (2) It fails to provide user anonymity; (3) It is prone to user impersonation attack. More recently, Li et al. found that Sood et al.'s dynamic ID-based authentication protocol for multi-server architecture is still vulnerable to several kinds of attacks and presented a new scheme that attempts to overcome the identified weaknesses. Notwithstanding their ambitions, Li et al.'s scheme is still found vulnerable to various known attacks by researchers. In this study, we perform a further cryptanalysis and uncover its two other vulnerabilities: (1) It cannot achieve user anonymity, which is the essential goal of a dynamic ID-based scheme; (2) It is susceptible to offline password guessing attack. The proposed cryptanalysis discourages any use of the two schemes under investigation in practice and reveals some subtleties and challenges in designing this type of schemes.

Keywords: Cryptanalysis, Authentication protocol, Offline password guessing attack, Smart card, Multi-Server architecture.

1 Introduction

With the rapid growth of Internet applications, the number of service providing servers proliferates at an ever-increasing rate [1, 2]. The distributed locations of service servers make it convenient and efficient for subscribers to access resources, and it is of great concern to protect the users and systems' security and privacy from malicious adversaries. Accordingly, user authentication is crucial to assure one communicating participant of the legitimacy of the corresponding

⋆ Corresponding author.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 462–475, 2012.

party by acquisition of corroborative evidence, preventing unauthorized clients from accessing system services for multi-server environment. Among numerous methods for user authentication, password based authentication using smart cards is the most convenient and effective two-factor authentication mechanism and has been widely adopted in many security-critical applications, such as e-banking, e-commerce and e-health [3].

In 1991, Chang and Wu [4] introduced the first password based remote user authentication schemes using smart cards, since then there have been many of this type of schemes proposed [5–8]. Although the issue of password authentication with smart cards for single-server environment recently has already been well studied in [5–8], it is extremely difficult for a user to remember these numerous different sets of identities and passwords when she employs these single-server architecture schemes to login and access different remote service servers.

To address this issue, a number of smart card based password authentication schemes for multi-server environment has been presented quite recently [9–12]. A sound and practical remote user authentication protocol for multi-server environment should be of high efficiency and can resist various related attacks, as well as the provision of some desirable features, such as mutual authentication, key agreement, local password update, user anonymity and so on. However, all of these schemes for multi-server environment are found impractical or completely insecure shortly after they were first proposed [13–15], which outlines the need for intensive further research and dynamic ID-based schemes that can preserve user anonymity are of particular interest.

In 2010, Shao and Chin [16] proposed an improved dynamic ID-based authentication scheme for multi-server environment to overcome the weakness of Hsiang-Shin's scheme [12]. The authors claimed that their improvement provides mutual authentication and is free from all related cryptographic attacks, such as offline password guessing attack, insider attack and impersonation attack. Although their scheme is efficient and superior to the previous solutions for implementation in resource-constrained applications, we find their scheme cannot achieve the claimed security: their scheme is vulnerable to offline password attack and user impersonation attack, and fails to preserve user anonymity.

More recently, Li et al. [17] pointed out that, besides a design flaw, Sood et al. scheme [15] is susceptible to leak-of-verifier attack and stolen smart card attack, and further proposed an efficient and secure dynamic ID-based authentication scheme using smart cards for multi-server architecture to cope with these identified problems. Unfortunately, just two months after Li et al.'s scheme was first published online, the replay attack, password guessing attack and masquerade attack are identified in their scheme by Han [18]. Later on, Xue et al. [19] also found Li et al.'s scheme cannot withstand the replay attack, denial of service attack, eavesdropping attack, internal attack and impersonation attack. Surprisingly, our further cryptanalysis demonstrates that Li et al.'s scheme still cannot preserve user anonymity, which is the most essential goal of a dynamic ID-based scheme. Besides, we also observed that Li et al.'s scheme is susceptible to another type of offline password guessing attack, which is more effective than and

different from Han's. In addition, we point out that Xue et al.'s improvement over Li et al.'s scheme is still vulnerable to a similar password guessing attack.

The remainder of this paper is organized as follows: in Section 2, we review Shao-Chin's scheme. Section 3 describes the weaknesses of Shao-Chin's scheme. Li et al.'s scheme is reviewed in Section 4 and the corresponding cryptanalysis is given is Section 5. Section 6 concludes the paper.

2 Review of Shao-Chin's Scheme

In this section, we examine the dynamic ID-based authentication scheme using smart cards proposed by Shao and Chin [16] in NSS 2010. Shao-Chin's scheme consists of five phases: registration phase, login phase, authentication phase, password change phase and track phase. For ease of presentation, we employ some intuitive abbreviations and notations listed in Table 1 and we will follow the notations in Shao-Chin's scheme as closely as possible.

Table 1. Notations

Symbol	Description
U_i	i^{th} user
S	remote server
ID_i	identity of user U_i
CID_i	dynamic identity of user U_i
P_i	password of user U_i
S_j	j^{th} service providing server
SID_j	identity of service server S_j
$\oplus$	the bitwise XOR operation
$\|$	the string concatenation operation
$h(\cdot)$	collision free one-way hash function
$A \rightarrow B : M$	message M is transferred through a common channel from A to B
$A \Rightarrow B : M$	message M is transferred through a secure channel from A to B

Besides the users and the service servers, there is another participant, called registration center (RC), involved in the system, and RC is trusted by all the users and service servers. Let x and z be two secret keys of RC.

2.1 Registration Phase

The registration phase is divided into two parts, namely, the server registration and the user registration.

(i) Server registration

1) S_j chooses her identity SID_j;
2) $S_j \Rightarrow RC : \{SID_j\}$;
3) RC computes $y_j = h(h(x) \parallel SID_j)$ and $h(z)$;
4) $RC \Rightarrow S_j : \{y_j, h(z)\}$.

(ii) User registration

1) U_i chooses her ID_i and P_i;
2) $U_i \Rightarrow RC : \{ID_i, P_i\}$;
3) RC computes $T_i = h(ID_i \parallel x)$, $R_i = h(x) \oplus h(z) \oplus T_i$, $V_i = T_i \oplus h(ID_i \parallel P_i)$ and $H_i = h(T_i)$ and stores $\{R_i, V_i, H_i, h(\cdot)\}$ in the smart card;
4) $RC \Rightarrow U_i$: A smart card containing security parameters $\{R_i, V_i, H_i, h(\cdot)\}$.

2.2 Login Phase

When U_i wants to login to S_j, the following operations will be performed:

Step L1. U_i inserts her smart card into card reader, and inputs ID_i and P_i.
Step L2. Smart card computes $T_i = V_i \oplus h(ID_i \parallel P_i)$, and checks whether H_i equals $h(T_i)$ or not. If they are equal, the user proceeds to the next step. Otherwise, the login request is rejected.
Step L3. Smart card generates a random number r and computes $B_1 = R_i \oplus T_i \oplus h(r \parallel T_i)$
Step L4. $U_i \rightarrow S_j : \{B_1\}$.
Step L5. On receiving B_1 from U_i, S_j computes $B_2 = B_1 \oplus h(z)$;
Step L6. $S_j \rightarrow U_i : \{B_2\}$.
Step L7. Smart card chooses a random number N_i and computes $y_j = h(B_2 \oplus h(r\|T_i)\|SID_j)$, $CID_i = ID_i \oplus h(B_2 \oplus h(r\|T_i)\|N_i)$, $G_i = CID_i \oplus h(y_j\|N_i)$ and $C = h(CID_i\|G_i\|N_i)$.
Step L8. $U_i \rightarrow S_j : \{C, G_i, N_i\}$.

2.3 Authentication Phase

After receiving the login request from U_i, S_j performs the following operations:

Step A1. The server S_j Computes $CID_i = G_i \oplus h(y_j\|N_i)$ and, then checks whether the received C is equal to the computed $h(CID_i\|G_i\|N_i)$. If the equality does not hold, the server S_j rejects the login request.
Step A2. S_j generates a random number N_j and computes $M_1 = h(CID \parallel SID_j \parallel N_i)$.
Step A3. $S_j \rightarrow U_i : \{M_1, N_j\}$.
Step A4. Upon receiving the response message from S_j, U_i computes $h(CID_i\| SID_j\|N_j)$ and compares it with M_1. The equality indicates the legitimacy of S_j. Otherwise, the login request is interrupted.
Step A5. U_i computes $M_2 = h(CID_i\|SID_j\|N_j)$.
Step A6. $U_i \rightarrow S_j : \{M_2\}$.
Step A7. On receiving M_2, S_j checks whether the received M_2 equals the computed $h(CID_i\|SID_j\|N_j)$. The equality indicates the legitimacy of U_i. Otherwise, the access request is interrupted.
Step A8. After authenticating each other, U_i and S_j use the same session key $SK = h(CID_i\|SID_j\|N_i\|N_j)$ to secure subsequent communications.

2.4 Password Change Phase and Track Phase

Since both the password change phase and track phase have little relevance with our discussions, they are omitted here.

3 Cryptanalysis of Shao-Chin's Scheme

There are three assumptions explicitly made in Shao-Chin's scheme [16]:

(i) An adversary $\mathcal{A}$ has total control over the communication channel between the user U and the remote server S. In other words, the attacker can intercept, block, delete, insert or alter any messages exchanged in the channel.
(ii) The secret parameters stored in the smart card can be revealed once a legitimate user's smart card is somehow obtained (e.g. picked up or stolen) by $\mathcal{A}$.
(iii) The user-memorable passwords are weak, i.e. of low entropy.

Note that the above three assumptions, which are also made in the latest works [5–8, 13–15], are indeed reasonable: (1) Assumption *i* is accordant with the common Dolev-Yao adversary model for distributed communication; (2) Assumption *ii* is practical when taking the state-of-the-art side-channel attack techniques [20–22] into consideration; and (3) Assumption *iii* reveals the reality that users are allowed to choose their own passwords at will during the password change phase and registration phase, usually the users are apt to choose passwords that are related to their personal life [23], such as meaningful dates, phone numbers or license plate numbers, and the human-memorable passwords tends to be "weak passwords" [24].

In the following discussions of the security pitfalls of Shao-Chin's scheme, based on the above three assumptions, we assume that an adversary can extract the secret parameters $\{V_i, R_i, H_i\}$ stored in the legitimate user's smart card, and could also intercept or block the exchanged messages $\{B_1, B_2, C, G_i, N_i, M_i, N_j, M_2\}$ during the login and authentication phase.

3.1 No Provision of User Anonymity

A protocol preserving user anonymity prevents an adversary from acquiring sensitive information about an individual's social circle, preferences, lifestyles, shopping patterns, etc. by analyzing the login history, the services requested, or the communications being accessed [25]. In addition, the leakage of user-specific information may cause an unauthorized entity or malicious attacker to track the user's current location and login history [26]. Hence, assuring anonymity not only does protect user privacy but also makes remote user authentication protocols more secure. In Shao-Chin's scheme, the dynamic-ID technique is employed to provide the feature of user anonymity, however, the following attack demonstrates the failure of their attempt.

Let us see how a dishonest service provider S_k colluding with a malicious privileged user U_m successfully breach the anonymity of any legitimate user, say U_i. U_m having her own smart card can gather information R_m, V_m from her own smart card, with previously intercepted authentication messages $\{B_1, B_2, N_i, G_i, C\}$ that are exchanged between U_m and any service provider, say S_j, U_m and S_k can collude to compute ID_i corresponding to U_i as follows:

Step 1. U_m computes $T_m = V_m \oplus h(ID_m \| P_m)$, where V_m is revealed from her own smart card, ID_m and P_m is known to herself;
Step 2. U_m computes $h(x) \oplus h(z) = R_m \oplus T_m$, where R_m is revealed;
Step 3. U_m and S_k collude to compute $h(x) = (h(x) \oplus h(z)) \oplus h(z) = (R_m \oplus T_m) \oplus h(z)$, where $h(z)$ is known to all service servers, including S_k.
Step 4. Guesses U_i's identity to ID_i^*;
Step 5. Computes $CID_i^* = ID_i^* \oplus h(h(x) \| N_i)$, where

$$\begin{aligned} h(x) &= h(x) \oplus (h(r\|T_i) \oplus h(r\|T_i)) \oplus (h(z) \oplus h(z)) \\ &= (h(x) \oplus h(z) \oplus h(r\|T_i)) \oplus (h(z) \oplus h(r\|T_i)) \\ &= B_1 \oplus (h(z) \oplus h(r\|T_i)) = B_2 \oplus h(r\|T_i) \end{aligned}$$

Step 6. Computes $C^* = h(CID_i^* \| G_i \| N_i)$, where G_i and N_i is intercepted.
Step 7. Verifies the correctness of ID_i^* by checking if the computed C^* is equal to the intercepted C;
Step 8. Goes back to Step 4 until the correct value of ID_i is found.

In practice, a user's identity is often drawn from a very limited space, say $\mathcal{D}_{id}$, the above procedure can be completed in polynomial time.

It is worth noting that, in the above attack, the malicious user U_m only needs to extract the security parameters stored in her own smart card, she does not need to obtain any information about the victim user U_i except the public authentication messages originating from U_i. As a result, the above attack is effective and practical. In conclusion, once an internal user colludes with a dishonest service server, user anonymity will be breached in Shao-Chin's scheme, while user anonymity is the most essential security feature that a dynamic identity-based authentication scheme is designed to provide.

3.2 Offline Password Guessing Attack

As stated in Section 3.1, any legitimate user U_i's identity can be breached when an internal malicious user U_m colludes with a service server S_k. Once the victim user U_i's identity ID_i is obtained by U_m and S_k, U_i's password P_i can also be offline guessed as follows:

Step 1. Guesses the value of P_i to be P_i^* from a dictionary space $\mathcal{D}_{pw}$.
Step 2. Computes $T_i^* = h(ID_i \| P_i^*) \oplus V_i$, where V_i is extracted from U_i's smart card.
Step 3. Verifies the correctness of P_i^* by checking if the computed $h(T_i^*)$ is equal to the revealed H_i.
Step 4. Repeats the above steps until the correct value of P_i is found.

Let $|\mathcal{D}_{id}|$ and $|\mathcal{D}_{pw}|$ denote the number of identities in identity space $\mathcal{D}_{id}$ and the number of passwords in password space $\mathcal{D}_{pw}$, respectively. The running time of the above attack procedure is $\mathcal{O}(|\mathcal{D}_{id}| * (3T_H + 5T_X) + |\mathcal{D}_{pw}| * (2T_H + T_X))$, where T_H is the running time for Hash operation and T_X is the running time for XOR operation. Since both password and identity are human-memorable short strings but not high-entropy keys, in other words, they are often chosen from two corresponding dictionaries of small size, e.g. $|\mathcal{D}_{id}| \leq |\mathcal{D}_{pw}| = 10^6$ [24]. As $|\mathcal{D}_{id}|$ and $|\mathcal{D}_{pw}|$ are very limited in practice, the above attack can be completed in polynomial time.

Note that, in this attack, the malicious user U_m not only needs to extract the security parameters stored in her own smart card, but also needs to obtain the secret data stored in the smart card of victim user U_i. Although this assumption is much constrained, our attack demonstrates the feasibility of offline password guessing attack on Shao-Chin's scheme under their non-tamper resistance assumption of the smart card, thereby contradicting the claim made in [16].

3.3 User Impersonation Attack

An internal malicious user U_m and a service server S_k can collude to impersonate any legitimate user (even non-existent user), say U_{ran}, to login any service server, say S_j, as follows:

Step 1. U_m computes $T_m = V_m \oplus h(ID_m \| P_m)$, as V_m is revealed from her own smart card, ID_m and P_m is known to herself;
Step 2. U_m computes $h(x) \oplus h(z) = R_m \oplus T_m$, where R_m is revealed;
Step 3. S_k and U_m collude to compute $h(x) = (h(x) \oplus h(z)) \oplus h(z) = (R_m \oplus T_m) \oplus h(z)$, where $h(z)$ is known to all service servers, including S_k.
Step 4. U_m sends a random value X to any service server, say S_j;
Step 5. U_m ignores the response $\{B_2\}$ sent back by S_j and computes $y_j = h(h(x) \| SID_j)$, where SID_j is S_j's identity.
Step 6. U_m computes $CID_{ran} = ID_{ran} \oplus h(h(x) \parallel N_{ran})$, $G_{ran} = CID_{ran} \oplus h(y_j \parallel N_{ran})$ and $C = h(CID_{ran} \parallel G_{ran} \parallel N_{ran})$, where N_{ran} is a random number chosen by U_m.
Step 7. U_m sends $\{C, G_{ran}, N_{ran}\}$ to S_j.
Step 8. On receiving the response $\{M_{ran}, N_j\}$ sent back by S_j, U_m computes $M_2 = h(CID_{ran} \parallel SID_j \parallel N_j)$ and the session key $SK = h(CID_{ran} \parallel SID_j \parallel N_{ran} \parallel N_j)$.
Step 9. U_m sends $\{M_2\}$ to S_j.

It is easy to see that: 1) On receiving X sent by U_m in Step 4, S_j will send back $B_2 = X \oplus h(z)$ according to the protocol; 2) On receiving $\{C, G_{ran}, N_{ran}\}$ sent by U_m in Step 7, S_j will find no abnormality when checking the validity of C, because U_m indeed has computed the correct $y_j = h(h(x)\|SID_j) = h(B_2 \oplus h(r\|T_i)\|SID_j)$ in Step 5, and the latter expression is justified as follows

$$\begin{aligned} h(x) &= h(x) \oplus (h(r\|T_i) \oplus h(r\|T_i)) \oplus (h(z) \oplus h(z)) \\ &= (h(x) \oplus h(z) \oplus h(r\|T_i)) \oplus (h(z) \oplus h(r\|T_i)) \\ &= B_1 \oplus (h(z) \oplus h(r\|T_i)) = B_2 \oplus h(r\|T_i). \end{aligned}$$

3) On receiving M_2 sent by U_m in Step 9, S_j will find no abnormality when checking the validity of M_2, because U_m has indeed computed the valid $CID_{ran} = ID_{ran} \oplus h(h(x) \parallel N_{ran})$ in Step 5, where $h(x) = B_2 \oplus h(r \| T_i)$.

It is worth noting that, as with the password guessing attack presented in Section 3.1, in this attack, the malicious user U_m only needs to extract the security parameters stored in her own smart card, she does not need to obtain any information about the victim U_i except the public authentication messages originating from U_i. As a result, this impersonation attack is effective and practical.

4 Review of Li et al.'s Scheme

In this section, we briefly review the dynamic identity based authentication protocol for multi-server architecture using smart cards proposed by Li et al. in 2012. Li et al.'s protocol also involves three participants, i.e., the user (U_i), the service providing server (S_j) and the control server (CS). It is should be noted that CS, a trusted party, is not only responsible for the registration but also involved in the authentication process of U_i and S_j. CS is in possession of a master secret key x and a secret number y. There are four phases in their protocol: registration, login, authentication and session key agreement, and password change. In the following, we employ the notations listed in Table 1.

4.1 Registration Phase

The registration phase can be divided into two parts, namely, the server registration and the user registration.

(i) Server registration
1) S_j chooses her identity SID_j;
2) $S_j \Rightarrow CS : \{SID_j\}$;
3) CS computes $h(SID_j \parallel y)$ and $h(x \parallel y)$;
4) $CS \Rightarrow S_j : \{h(x \parallel y), h(SID_j \parallel y)\}$.

(ii) User registration

1) U_i freely chooses her ID_i and P_i, and chooses a random number b. Then, U_i computes $A_i = h(b \parallel P_i)$;
2) $U_i \Rightarrow CS : \{ID_i, A_i\}$;
3) CS computes $B_i = h(ID_i \parallel x)$, $C_i = h(ID_i \parallel h(y) \parallel A_i)$, $D_i = B_i \oplus h(ID_i \parallel A_i)$, $E_i = B_i \oplus h(y \parallel x)$, and stores $\{C_i, D_i, E_i, h(\cdot), h(y)\}$ in the smart card;
4) $CS \Rightarrow U_i$: A smart card containing parameters $\{C_i, D_i, E_i, h(\cdot), h(y)\}$.
5) Upon receiving the smart card, U_i enters b into it.

4.2 Login Phase

When U_i wants to login to S_j, the following operations will be performed:

Step L1. User U_i inserts her smart card into a card reader and inputs her identity ID_i, password P_i and the service server's identity SID_j.
Step L2. The smart card computes $A_i = h(b \parallel P_i)$ and $C_i' = h(ID_i \parallel h(y) \parallel A_i)$, and checks whether $C_i' = C_i$. If they are equal, it indicates that U_i is a legal card holder.
Step L3. The smart card generates a random number N_{i1}, and computes $B_i = D_i \oplus h(ID_i \parallel A_i)$, $F_i = h(y) \oplus N_{i1}$, $P_{ij} = E_i \oplus h(h(y) \parallel N_{i1} \parallel SID_j)$, $CID_i = A_i \oplus h(B_i \parallel F_i \parallel N_{i1})$, $G_i = h(B_i \parallel A_i \parallel N_{i1})$.
Step L4. $U_i \rightarrow S_j : \{F_i, G_i, P_{ij}, CID_i\}$.

4.3 Authentication and Session Key Agreement Phase

Step A1. On receiving the login request, S_j chooses a random number N_{i2}, and computes $K_i = h(SID_j \parallel y) \oplus N_{i1}$ and $M_i = h(h(x \parallel y) \parallel N_{i2})$.
Step A2. $S_j \rightarrow CS : \{F_i, G_i, P_{ij}, CID_i, SID_j, K_i, M_i\}$.
Step A3. Upon receiving the login request $\{F_i, G_i, P_{ij}, CID_i, SID_j, K_i, M_i\}$, CS computes $N_{i2} = K_i \oplus h(SID_j \parallel y)$, $M_i' = h(h(x\|y)\|N_{i2})$, and checks whether M_i' equals the received M_i. If they are equal, the validity of the server S_j is verified by the control server CS. Otherwise, the CS terminates the session.
Step A4. CS computes $N_{i1} = F_i \oplus h(y)$,$B_i = P_{ij} \oplus h(h(y)\|N_{i1}\|SID_j) \oplus h(y\|x)$ $(= E_i \oplus h(y\|x))$, $A_i = CID_i \oplus h(B_i \parallel F_i \parallel N_{i1})$, $G_i' = h(B_i \parallel A_i \parallel N_{i1})$ and checks $G_i' \stackrel{?}{=} G_i$. If the verification holds, the legitimacy of user U_i is authenticated by CS. Otherwise CS terminates the session.
Step A5. CS generates a random number N_{i3}, and computes $Q_i = N_{i1} \oplus N_{i3} \oplus h(SID_j \parallel N_{i2})$,$R_i = h(A_i \parallel B_i) \oplus h(N_{i1} \oplus N_{i2} \oplus N_{i3})$, $V_i = h(h(A_i \parallel B_i) \parallel h(N_{i1} \oplus N_{i2} \oplus N_{i3}))$, $T_i = N_{i2} \oplus N_{i3} \oplus h(A_i \parallel B_i \parallel N_{i1})$.
Step A6. $CS \rightarrow S_j : \{Q_i, R_i, V_i, T_i\}$.
Step A7. On receiving the authentication message $\{Q_i, R_i, V_i, T_i\}$ from CS, server S_j computes $N_{i1} \oplus N_{i3} = Q_i \oplus h(SID_j \parallel N_{i2})$, $h(A_i \parallel B_i) = R_i \oplus h(N_{i1} \oplus N_{i3} \oplus N_{i2})$, $V_i' = h(h(A_i \parallel B_i) \parallel h(N_{i1} \oplus N_{i3} \oplus N_{i2}))$, and checks $V_i' \stackrel{?}{=} V_i$. If they are not equal, S_j terminates the session. Otherwise, the legitimacy of CS is authenticated by the server S_j.
Step A8. $S_j \rightarrow U_i : \{V_i, T_i\}$.
Step A9. Upon receiving $\{V_i, T_i\}$ from S_j, the smart card computes $N_{i2} \oplus N_{i3} = T_i \oplus h(A_i \parallel B_i \parallel N_{i1})$, $V_i' = h(h(A_i \parallel B_i) \parallel h(N_{i2} \oplus N_{i3} \oplus N_{i1}))$, and checks $V_i' \stackrel{?}{=} V_i$. If the verification fails, the user U_i terminates the session. Otherwise, the legitimacy of the control server CS and the server S_j is authenticated by user U_i.

Finally, the user U_i, the server S_j and the control server CS agree on a common session key $SK = h(h(A_i \parallel B_i) \parallel (N_{i1} \oplus N_{i2} \oplus N_{i3}))$.

4.4 Password Change Phase

This phase is performed locally. When the user wants to update her password, this phase is invoked. Since this phase has little relevance with our discussions, it is omitted here.

5 Cryptanalysis of Li et al.'s Scheme

The three assumptions presented in Section 3 is also explicitly made in Li et al.'s paper when they analyze the security of Sood et al.'s scheme, and thus our following cryptanalysis is also based on these three assumptions.

Although Li et al.'s scheme has many attractive properties, such as provision of local password change, high efficiency and no time-synchronization problem, it fails to achieved many of the claimed security goals and has been found vulnerable to replay attack, password guessing attack and user impersonation attack by Han [18]. Besides these security pitfalls, later on Xue et al. further found it prone to leak-of-verifier attack, server spoofing attack and denial of service attack,[1] and they also presented an improvement.

Surprisingly, our further cryptanalysis demonstrates that Li et al.'s scheme still cannot preserve user anonymity, which is the most crucial goal of a dynamic ID-based scheme. Besides, we also observe that Li et al.'s scheme is susceptible to another type of offline password guessing attack, which is more effective than and different from Han's. Furthermore, we point out that Xue et al.'s improvement over Li et al.'s scheme is still vulnerable to a similar password guessing attack.

5.1 No Provision of User Anonymity

Let us see how a dishonest service provider S_k colluding with a malicious internal user U_m successfully breach the anonymity of any legitimate user, say U_i. U_m having her own smart card can gather information $h(y)$ from her own smart card, with previously intercepted authentication messages $\{P_{ij}, SID_j\}$ that are exchanged between U_m, CS and any service provider, say S_j, U_m and S_k can collude to compute E_i corresponding to any user U_i as follows:

Step 1. U_m extracts $h(y)$ from her own smart card;
Step 2. U_m and S_k collude to compute $N_{i1} = F_i \oplus h(y)$, where F_i is intercepted from the public channel;
Step 3. U_m and S_k collude to compute $E_i = P_{ij} \oplus h(h(y) \parallel N_{i1} \parallel SID_j)$, where P_{ij} and SID_j are intercepted from the public channel.

As E_i is kept the same for all the login requests of user U_i and is specific to U_i, this E_i can be seen as user U_i's identification. And an adversary can, therefore, use this information to identify and trace U_i's login requests and activities. By

[1] We think Xue et al.'s internal attack and eavesdropping attack only constitute parts of replay attack, server spoofing attack, etc, and they may not be considered as independent kinds of attacks, and thus they are not listed here.

generalizing the above attack, any legal user who logins to service servers would be exposed to U_m and S_k, and thus user anonymity is not preserved.

It should be noted that, in the above attack, the malicious user U_m only needs to extract the security parameters stored in her own smart card, she does not need to obtain any information about the victim user U_i except the public authentication messages originating from U_i. As a result, the above attack is effective and practical. In conclusion, once an internal user colludes with a dishonest service server, user anonymity will be breached in Li et al.'s scheme, while user anonymity is the most crucial security feature that a dynamic identity-based authentication scheme is designed to provide.

5.2 Offline Password Guessing Attack

Let us consider the following scenarios. In case a legitimate user U_i's smart card is stolen by a malicious internal user U_m, and the stored secret values $h(y)$, D_i, E_i and b can be extracted. Note that this assumption is reasonable as described in Assumption *iii* and it is also explicitly made in Li et al.'s scheme. With the previously eavesdropped message $\{F_i, CID_i, G_i\}$, this malicious internal user U_m can successfully guess the password of U_i as follows:

Step 1. Extracts $h(y)$ from her own smart card;
Step 2. Computes $N_{i1} = F_i \oplus h(y)$, where F_i is intercepted from the public channel;
Step 3. Computes $E_i = P_{ij} \oplus h(h(y) \parallel N_{i1} \parallel SID_j)$, where P_{ij} and SID_j are intercepted from the public channel.
Step 4. Computes $h(y\|x) = E_m \oplus B_m = E_m \oplus D_m \oplus h(ID_m\|A_m) = E_m \oplus D_m \oplus h(ID_m\|h(b\|P_m))$, where E_m, D_m and b are revealed from U_m's own smart card;
Step 5. Computes $B_i = E_i \oplus h(y\|x)$, where E_i is revealed from U_i's smart card;
Step 6. Computes $A_i = CID_i \oplus h(B_i\|F_i\|N_{i1})$;
Step 7. Guesses the value of P_i to be P_i^* from the password space $\mathcal{D}$.
Step 8. Computes $A_i^* = h(b\|P_i^*)$, where b is revealed from U_i's smart card.
Step 9. Verifies the correctness of P_i^* by checking if A_i^* equals to A_i.
Step 10. Repeats Steps 7, 8 and 9 until the correct value of P_i is found.

Let $|\mathcal{D}|$ denote the number of passwords in the password space $\mathcal{D}$. Then the running time of the attacker U_m is $\mathcal{O}(|\mathcal{D}| * (5T_H + 6T_X))$, where T_H is the running time for Hash operation and T_X is the running time for XOR operation. So, the time for U_m to recover the password is a linear function of the number of passwords in the password space. When the password space is small, e.g., $|\mathcal{D}| = 10^6$ [24], U_m may recover the password in seconds on a PC.

It should be noted that, in this attack, the malicious user U_m only needs to guess U_i's password, while in the offline password guessing attack proposed by Han [18], the attacker needs to guess both U_i's password and identity correctly at the same time. From this point of view, our attack is more effective. But our disadvantage is that, the adversary in our attack should be an internal user, while the adversary in Han's attack is not subject to this restriction.

5.3 Offline Password Guessing Attack on Xue et al.'s Improvement

In [19], Xue et al. pointed out that Li et al.'s scheme vulnerable to several attacks and further proposed an improvement that is claimed to be secure.[2] However, we find Xue et al.'s improvement is still vulnerable to an offline password guessing attack as described in the following.

Let us consider the following scenarios. In case a legitimate user U_i's smart card is stolen by an adversary $\mathcal{A}$, and the stored secret values such as C_i, D_i and b can be extracted. Note that this assumption is explicitly made in Xue et al.'s improvement. With a previously eavesdropped message $\{F_i, PID_i, TS_i\}$, $\mathcal{A}$ can acquire U_i's password PW_i by performing the following attack procedure:

Step 1. Guesses the value of P_i to be P_i^* from the password space $\mathcal{D}$;
Step 2. Computes $A_i^* = h(b\|P_i^*)$, where b is revealed from U_i's smart card;
Step 3. Computes $B_i^* = D_i \oplus h(PID_i \oplus A_i^*)$, where PID_i is intercepted from the public channel;
Step 4. Computes $N_{i1}^* = F_i \oplus B_i^*$;
Step 5. Computes $G_i^* = b \oplus h(B_i^*\|N_{i1}^*\|TS_i\|\text{“11”})$;
Step 6. Verifies the correctness of P_i^* by checking if G_i^* equals to the intercepted G_i;
Step 7. Repeats the above steps until the correct value of P_i is found.

Since the size of password dictionary, i.e.$|\mathcal{D}|$, often is very limited in practice, the above attack procedure can be completed in polynomial time.

Notes and Countermeasure. We have analyzed more than sixty recently proposed smart card based password authentication schemes for single-server environment and twelve schemes for multi-server architecture, and find these schemes (no matter for single-server environment or multi-server architecture) that do not employ public-key techniques definitely vulnerable to the offline password guessing attack under the three assumptions (most essentially, the non-tamper resistance assumption of the smart card) introduced in Section 3. In other words, all these schemes that do not employ public-key techniques but claim to be secure under these three assumptions are found problematic. A related work done by Halevi and Krawczyk [27] provides very strong evidence (with the probability of $\boldsymbol{P \neq NP}$) that, under the common Dolev-Yao adversary model, no password protocol (the traditional one-factor password authentication) can be free from offline password guessing attack if the public-key techniques are not employed. Here, we conjecture that under the three assumptions introduced in Section 3, no smart card based password protocol (two-factor authentication) can be free from offline password guessing attack if the public-key techniques are not employed. And now the countermeasure is obvious: resorting to public-key techniques like [5–8].

[2] According to Xue et al.'s statement, this improvement has been submitted to Journal of Network and Computer Applications.

6 Conclusion

Understanding security failures of cryptographic protocols is the key to both revising existing protocols and proposing future schemes. In this paper, we have shown that two dynamic ID-based remote user authentication schemes for multi-server environment are completely broken and only radical revisions of the protocols can possibly eliminate the identified defects and thus the two schemes under investigation are not recommended for practical application. Remarkably, our cryptanalysis highlights the difficulties and challenges in designing secure and efficient dynamic ID-based remote user authentication schemes for multi-server architecture, in the hope that no similar mistakes are made in the future.

Acknowledgment. This research was partially supported by the National Natural Science Foundation of China (NSFC) under No. 61170241 and the International Exchange Program of Harbin Engineering University for Innovation-oriented Talents Cultivation.

References

1. Bouyoucef, K., Khorasani, K.: A robust distributed congestion-control strategy for differentiated-services network. IEEE Transactions on Industrial Electronics 56(3), 608–617 (2009)
2. Barolli, L., Xhafa, F.: JXTA-OVERLAY: A P2P platform for distributed, collaborative and ubiquitous computing. IEEE Transactions on Industrial Electronics 58(6), 2163–2172 (2010)
3. Lin, S., Hung, M., Tsai, C., Chou, L.: Development of an ease-of-use remote healthcare system architecture using rfid and networking technologies. Journal of Medical Systems, 1–15 (2012), doi:10.1007/s10916-012-9836-0
4. Chang, C.C., Wu, T.C.: Remote password authentication with smart cards. IEE Proceedings-Computers and Digital Techniques 138(3), 165–168 (1991)
5. Ma, C.-G., Wang, D., Zhang, Q.-M.: Cryptanalysis and Improvement of Sood et al.'s Dynamic ID-Based Authentication Scheme. In: Ramanujam, R., Ramaswamy, S. (eds.) ICDCIT 2012. LNCS, vol. 7154, pp. 141–152. Springer, Heidelberg (2012)
6. Wu, S.H., Zhu, Y.F., Pu, Q.: Robust smart-cards-based user authentication scheme with user anonymity. Security and Communication Networks 5(2), 236–248 (2012)
7. Wang, D., Ma, C.G., Wu, P.: Secure Password-Based Remote User Authentication Scheme with Non-tamper Resistant Smart Cards. In: Cuppens-Boulahia, N., Cuppens, F., Garcia-Alfaro, J. (eds.) DBSec 2012. LNCS, vol. 7371, pp. 114–121. Springer, Heidelberg (2012)
8. Wang, Y.: Password Protected Smart Card and Memory Stick Authentication against Off-Line Dictionary Attacks. In: Gritzalis, D., Furnell, S., Theoharidou, M. (eds.) SEC 2012. IFIP AICT, vol. 376, pp. 489–500. Springer, Heidelberg (2012)
9. Lin, I., Hwang, M., Li, L.: A new remote user authentication scheme for multi-server architecture. Future Generation Computer Systems 19(1), 13–22 (2003)
10. Tsaur, W., Wu, C., Lee, W.: A smart card-based remote scheme for password authentication in multi-server internet services. Computer Standards & Interfaces 27(1), 39–51 (2004)
11. Liao, Y., Wang, S.: A secure dynamic id based remote user authentication scheme for multi-server environment. Computer Standards & Interfaces 31(1), 24–29 (2009)

12. Hsiang, H., Shih, W.: Improvement of the secure dynamic id based remote user authentication scheme for multi-server environment. Computer Standards & Interfaces 31(6), 1118–1123 (2009)
13. Tan, Z.: Cryptanalysis of two id based password authentication schemes for multi-server environments. International Journal of Digital Content Technology and its Applications 5(1), 87–94 (2011)
14. Yeh, K., Lo, N., Li, Y.: Cryptanalysis of hsiang-shihs authentication scheme for multi-server architecture. International Journal of Communication Systems 24(7), 829–836 (2011)
15. Sood, S., Sarje, A., Singh, K.: A secure dynamic identity based authentication protocol for multi-server architecture. Journal of Network and Computer Applications 34(2), 609–618 (2011)
16. Shao, M., Chin, Y.: A novel approach to dynamic id-based remote user authentication scheme for multi-server environment. In: 2010 4th International Conference on Network and System Security (NSS 2010), pp. 548–553. IEEE Press, New York (2010)
17. Li, X., Xiong, Y., Ma, J., Wang, W.: An enhanced and security dynamic identity based authentication protocol for multi-server architecture using smart cards. Journal of Network and Computer Applications 35(2), 763–769 (2012)
18. Han, W.: Weaknesses of a dynamic identity based authentication protocol for multi-server architecture. Arxiv preprint arXiv:1201.0883 (2012), http://arxiv.org/abs/1201.0883
19. Xue, K., Hong, P., Ma, C.: A lightweight dynamic pseudonym identity based authentication and key agreement protocol without verification tables for multi-server architecture. Arxiv preprint arXiv:1204.3831 (2012), http://arxiv.org/abs/1204.3831
20. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–789. Springer, Heidelberg (1999)
21. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Examining smart-card security under the threat of power analysis attacks. IEEE Transactions on Computers 51(5), 541–552 (2002)
22. Kasper, T., Oswald, D., Paar, C.: Side-Channel Analysis of Cryptographic RFIDs with Analog Demodulation. In: Juels, A., Paar, C. (eds.) RFIDSec 2011. LNCS, vol. 7055, pp. 61–77. Springer, Heidelberg (2012)
23. Florencio, D., Herley, C.: A large-scale study of web password habits. In: Proceedings of WWW 2007, pp. 657–666. ACM Press, New York (2007)
24. Klein, D.V.: Foiling the cracker: A survey of, and improvements to, password security. In: Proceedings of the 2nd USENIX Security Workshop, pp. 5–14 (1990)
25. Bao, F., Deng, R.: Privacy Protection for Transactions of Digital Goods. In: Qing, S., Okamoto, T., Zhou, J. (eds.) ICICS 2001. LNCS, vol. 2229, pp. 202–213. Springer, Heidelberg (2001)
26. Tang, C., Wu, D.: Mobile privacy in wireless networks-revisited. IEEE Transactions on Wireless Communications 7(3), 1035–1042 (2008)
27. Halevi, S., Krawczyk, H.: Public-key cryptography and password protocols. ACM Transactions on Information and System Security 2(3), 230–268 (1999)

A Secure and Private RFID Authentication Protocol under SLPN Problem

Mohammad S.I. Mamun, Atsuko Miyaji, and Mohammad S. Rahman

Japan Advanced Institute of Science and Technology (JAIST)
Ishikawa, Japan
{mamun,miyaji,mohammad}@jaist.ac.jp

Abstract. Authentication is one of the prominent features of RFID system. As wireless link between the tag and the reader in an RFID system is vulnerable against active adversary, ample research has been done in this area. In this paper[1], we present a novel, efficient and privacy preserving mutual authentication protocol of HB-family to meet the demand of low-cost tags. It is composed of Subspace Learning Parity from Noise problem (SLPN) and pseudo-inverse matrix properties; both of them significantly minimize the cost in terms of computation and hardware requirements. We compare our protocol with other existing HB and non-HB authentication protocols according to their construction and achievements of security and privacy attributes.

1 Introduction

RFID system simplifies data acquisition and management with an automated identification of an object to which the tag is attached. A number of authentication schemes have been proposed targeting privacy, security, efficiency and performance issues. Majority of the authentication protocols in this area use symmetric key ciphers, as asymmetric key ciphers are too expensive for a compact hardware like RFID tag. For example, RSA require more than 30,000 gates, which is too expensive for low-cost tag where maximum 2,000 gates out of 10,000 gates are available for security purpose [1].

LPN problem is a light-weight provably-secure cryptographic scheme which was first introduced in 2001 by Hopper & Blum [3]. LPN based authentication is not only *theoretically secure* in terms of provable security, but also provides better *efficiency* than classical symmetric ciphers that are not related to hard problems. There has been a large body of research on HB protocol that outputs protocols such as HB^{+}, HB^{++}, $HB^{\#}$, HB-MP, HB-MP^{+}, HB^{*} etc.[5–9, 11, 22]. Unfortunately all of them were later shown insecure or susceptible to particular attacks [10, 11]. In [2], authors propose an authentication protocol based on Subspace LPN (SLPN) problem with tight security reduction which is as efficient as the previous HB-family but has twice the key length; in addition their proof works in a quantum setting which leads the protocol secure against quantum adversaries.

[1] Research supported by Graduate Research Program (GRP), JAIST foundation grants.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 476–489, 2012.

To the best of our knowledge, the latest addition to HB-family for RFID authentication is F-HB, where authors use 2 LPN problem as their basic computation [17]. We carefully observe that the Toeplitz matrix multiplication (EX-OR operation) for multiple bit LPN problem and MAC generation in the main protocol of [17] are not consistent with matrix size, although authors did not clarify specific matrix size in operation; and threshold value for LPN problem is not specified concretely. Moreover, in the last protocol transcripts, where a tag's secret key is updated, *if-checking* is not consistent and is not based on LPN problem; but an EX-OR vector computation. Unlike [17], our protocol follows SLPN based problem for tag authentication where secret key is not a vector but a binary matrix. In addition, we introduce pseudo-inverse matrix for updating the secret key of the tag and apply SLPN problem for both the tag and the reader authentication. As a consequence, our proposed protocol is more robust against quantum adversaries while efficient like previous HB-protocol family.

The rest of this paper is organized as follows. Section 2 introduces notations and assumptions used in this paper and other useful definitions related to basic primitives and security notions. The proposed protocol is described in Section 3. In Section 4, all achieved security and privacy attributes are discussed in detail with their proof; while Section 5 covers the analysis and comparison results. Finally, Section 6 concludes this paper.

2 Preliminaries

In this section, we first briefly introduce the notations used in the paper in Table 1. Then we discuss some inevitable assumptions followed by useful definitions for primitives and security notions.

2.1 Assumption

RFID system described in this paper consists of a single legitimate reader and a set of tags (EPC global Class 1 generation 2). Reader is connected to the back-end server that stores all the relevant data including tag database. Each tag has its unique identification T_{id} and session key S_i. T_{id} is used as the shared secret key between the tag and the reader.

The authentication protocol is an interactive protocol executed between *tags/prover* and a *reader/verifier* where both of them are *probabilistic polynomial time* (PPT) algorithm. All communications between the server and the reader are assumed to be secure and over authentic channel. For simplicity, we consider reader and server as identical. Throughout the paper, we use the term reader and server interchangeably. A tag is not tamper-resistant device; so its session key S_i is refreshed after each session completes successfully. For updating key, tag authenticates the reader first. Adversary cannot compromise reader/server and it cannot corrupt the tag until it compromises both T_{id} and S_i at a time. However, if both of the secret keys are exposed at a time, the adversary can trace the tag for certain period i until the next authentication cycle starts.

Table 1. Notations used in the paper

λ	security parameter
$\mathbb{Z}_p$	set of integers modulo an integer $p \geq 1$
$l \in \mathbb{N}$	length of the secret key
$n \in \mathbb{N}$	number of parallel repetitions $n \leq l/2$
T_{id}	$2l$ bit EPC or unique ID of a tag
I_i	nIndex of the tag during time period i
P_i	$l \times l$ bit matrices as session key for the reader during time period i
S_i	$l \times n$ bit matrices as session key between the reader and the tag during time period i
s	$2l$ bit vector random binary number generated by reader.
s'	$2l$ bit vector random binary number generated by tag
$w(s)$	Hamming weight of the vector s
τ	Parameter of the Bernoullli error distribution $\mathbf{Ber}_\tau$ where $\tau \in]0, 1/2[$
τ'	Authentication verifier acceptance threshold (Tag/Reader) where $\tau' = 1/4 + \tau/2$
e	n bit vector from Bernoullli distribution $\mathbf{Ber}_\tau$ with parameter τ ; $Pr[e = 1] = \tau$
$[Q]$	$l \times n$ bit randomly generated non-singular binary matrices by reader
$[S]^T$	transpose of matrices $[S]$ i.e., $T : \mathbb{Z}_2^{n \times l} \rightarrow \mathbb{Z}_2^{l \times n}$
$[P]^+$	pseudo-inverse of a matrices $[P]$
$(x_{\downarrow} y)$	the vector derived from x by deleting all the bits $x[i]$ where $y[i] = 0$
$\oplus, \|$	bitwise XOR operation and concatenation of two vectors respectively

We assume tag binary identification T_{id} is unique within an RFID system. To avoid exhaustive database search at the reader hash-index (I) is used. Database at the server associates the tag index with other tag related data e.g., T_{id}, S_i, P_i etc.

2.2 Definitions for Primitives

Definition 1. *The **LPN problem** is to distinguish from random binary vectors*[2] *with a random "secret" vector. Let* $[R] \in_R \mathbb{Z}_2^{l \times n}$, $s \in_R \mathbb{Z}_2^n$, τ *be noise parameters, and* $e \in \mathbb{Z}_2$ *selected from* Ber_τ *s.t.* $w(e) \leq \tau l$. *Given* $r = ([R]^T \cdot s) \oplus e \in \mathbb{Z}_2^l$, *finding* $s' \in_R \mathbb{Z}_2^n$ *such that* $w([R]^T \cdot s') \oplus r) \leq \tau l$ *is denoted by the distribution* $\mathcal{D}_{\tau,l}(s')$. *The* $LPN_{\tau,l}$ *problem is to distinguish an oracle returning samples from* $\mathcal{D}_{\tau,l}(s')$, *or an oracle returning uniform samples* U_l.

Definition 2. *The **Subspace LPN (SLPN)** problem is defined as a biased half-space distribution where adversary can ask not only with secret "s" but also with* $r'.s \oplus e'$; *where* $\mathbf{e}', \mathbf{r}'$ *can be adaptively chosen with sufficient* $rank(r')$. *Let* $s \in \mathbb{Z}_2^l$ *and* $l, n \in \mathbb{Z}$ *where* $n \leq l$. *Decisional Subspace LPN problem (SLPN) is* (t, Q, ϵ)*-hard such that*

$$Adv_{\mathcal{A}}^{\mathbf{SLPN}}(\tau, l, n) = Pr[LPN_{\tau,l,n}(s, \cdot, \cdot) = 1] - Pr[U_l : LPN_{1/2}(\cdot, \cdot) = 1] \leq \epsilon$$

[2] Result of noisy inner products of vectors.

Definition 3. *The **Subset LPN problem** (**SLPN***) is defined as a weaker version of SLPN problem where the adversary cannot ask for all inner products with $r' \cdot s \oplus e'$; for any $rank(r') \geq n$ but only with **subset** of s. Let $(l, n, v) \in \mathbb{Z}$ where $n \leq l$ and $w(v) \geq n$ where v can be adaptively chosen. Hence, $LPN^*_{\tau,l,n}(s, v)$ samples are of the form $([R]^T{}_{\downarrow} v \cdot s_{\downarrow} v) \oplus e$ and $LPN_{1/2}(v)$ takes v as input and output a sample of U_l. $SLPN^*$ problem is (t, Q, ϵ)-hard such that*

$$\mathbf{Adv}^{\mathbf{SLPN}^*}_{\mathbf{A}}(\tau, l, n) = Pr[LPN^*_{\tau,l,n}(s, \cdot) = 1] - Pr[U_l : LPN_{1/2}(\cdot) = 1] \leq \epsilon$$

Definition 4. *In linear algebra, a **pseudo-inverse** A^+ of a matrix A is a generalization of the inverse matrix. The most widely known and popular pseudo-inverse is the **MooreCPenrose** pseudo-inverse, which was independently described by E. H. Moore [12]. An algorithm for generating pseudo-random matrix on non-singular matrix $\mathbb{Z}_2$ is given in [13]. However, the matrix A is the unique matrix that satisfies the following properties: $AA^+A = A$, $A^+AA^+ = A^+$, $(A^+A)^T = A^+A$, $(A^+)^+ = A$, $(A^T)^+ = (A^+)^T$, $(A^+)^T = (A^T)^+$, $(AA^+)^T = AA^+$ where $T : \mathbb{Z}_2^{n \times l} \to \mathbb{Z}_2^{l \times n}$, $A^+ = (A^TA)^{-1}A^T$, and $A^+ = A^T(AA^T)^{-1}$ where $row(A)$ and $col(A)$ are linearly independent.*

2.3 Definitions for Security Notions

Definition 5. *A protocol is secure against **passive attacks**, if there exists no PPT adversary $\mathcal{A}$ that can forge the verifying entity with non-negligible probability by observing any number of interactions between the tag and reader.*

Definition 6. *A protocol (t, Q, ϵ) is called secure against **active attacks**, if there exists no PPT adversary $\mathcal{A}$, usually called $(\mathbf{Q}, \mathbf{t})$ **adversary**, running in time t and making Q queries to the honest prover, has probability more than ϵ. Adversary $\mathcal{A}$ runs in two stages: First, it observes and interrupts all the interactions between the target tag T and legitimate reader with concurrent executions according to the defined security. Then it is allowed only one time to convince the reader. Note that, this time $\mathcal{A}$ is not allowed to continue his attacks in time instance t; but can utilize several discrete or successive time period.*

Definition 7. *In **man-in-the-middle** (MIM) attack, adversary $\mathcal{A}$ is allowed to maintain connections with both the tag and the reader, making the tag believe that they are talking directly to the reader over a secure connection, when in fact the entire communication is controlled by $\mathcal{A}$. Then $\mathcal{A}$ interacts with the reader to authenticate. The goal of the attacker $\mathcal{A}$ is to authenticate successfully in Q rounds. $\mathcal{A}$ is successful iff it gets accept response from all Q rounds.*

Definition 8. ***Forward security** property means that even if the adversary obtains the current secret key, it cannot derive the keys used for past time periods.*

Definition 9. ***Backward security** is the opposite to the forward security. If the adversary can explore the secret of the tag at time i, it cannot be traced in future using the same secret. In other words, exposure of a tag's secret should not*

reveal any secret information regarding the future of the tag. But if an adversary is allowed to obtain full access to the tag's secret and thus can trace the target tag at least during the current session of authentication immediately following the attack, its not making any sense to perfect security in practice. Therefore, it is impossible to provide backward security for RFID-like device practically.

Definition 10. ***Tracking a tag*** *refers the attacker could guess the tag identity or link multiple authentication sessions of the same tag. In our protocol, adversary cannot recover S_i or any other information identifying that particular tag.*

Definition 11. *In* ***De-synchronization attack****, adversary aims to disrupt the key update leaving the tag and the reader in a desynchronized state and renders future authentication impossible.*

Definition 12. ***Denial of Service*** *(DoS) is an attempt to make a tag unavailable to its intended users. DoS resistance capability of the protocol is infinite as tag updates the key after reader authentication is successful.*

Definition 13. ***Tag cloning*** *entails that the data on a valid tag is scanned and copied by a malicious RFID reader and later the copied data will be embedded onto a fake tag.*

Definition 14. *In* ***replay attack****, an adversary reuses the communication scripts from the former sessions to perform a successful authentication between each tag and their reader.*

Definition 15. *An RFID system is said to be (Q, t, ϵ)* ***strong private****, if there exist no (Q, t) adversary $\mathcal{A}$ who can break its strong privacy with advantage $Adv^b_{\mathcal{A}}(k) \geq \epsilon$.*

3 Construction

We adopt the idea of key-insulation to slightly twist our 3-round mutual authentication protocol described in Fig. 1. Protocol allows significantly less computations to a tag. On the other hand, the most expensive computations of the protocol are handled by the reader. We use only random generation, bitwise XOR and matrix multiplication as tag operation. Protocol uses $(\lambda, \tau, \tau', n, l)$ as public parameters, where (τ, τ') are constant while (l, n) depends on the security parameter λ. For initialization, server generates initial index I_0, session key S_0 and its corresponding P_0 and other public parameters; and set necessary data into tag non-volatile memory. Note that, we use *matrix* as a secret, not a *vector*. Therefore, for each tag, there is a tuple $[I_i, T_{id}, S_{i-1}, S_i, P_{i-1}, P_i]$ to be stored in the back-end database of the server at any time instance i.

For *tag* authentication, let a tag have S_i and I_i, which have been derived from the previous $(i-1)$ successful authentication sessions.

Reader $(I_i, T_{id} \in \mathbb{Z}_2^{2l}, \mathbf{S_i} \in \mathbb{Z}_2^{l\times n}, \mathbf{P_i} \in \mathbb{Z}_2^{l\times l})$	**Tag**$(I_i, T_{id} \in \mathbb{Z}_2^{2l}, \mathbf{S_i} \in \mathbb{Z}_2^{l\times n})$
$s \in_R \mathbb{Z}_2^{2l}$; where $\mathbf{w}(s) = l$	
$\xrightarrow{\mathbf{s}}$	
	if $\mathbf{w}(s) \neq l$ **return**;
	$\mathbf{e} \in_R \mathbf{Ber}_\tau^n$;
	$\mathbf{r} := [\mathbf{S_i}]^{\mathbf{T}}.\ (T_{id\downarrow} s) \oplus \mathbf{e}$
	$s' \in_R \mathbb{Z}_2^{2l}$; where $\mathbf{w}(s') = l$
	$I_{i+1} = r$
	$\xleftarrow{(\mathbf{I_i}, \mathbf{s'}, \mathbf{r})}$
Lookup T_{id} by using hash-table index:	
Direct match: $I = I_i$; if (not found) then	
Brute-force search: find an entry $[I_i, T_{id}, \mathbf{S_{i-1}}, \mathbf{S_i}, \mathbf{P_{i-1}}, \mathbf{P_i}]$	
s.t., $\exists$ ($\mathbf{S_i}$ or, $\mathbf{S_{i-1}}$), for which the following satisfies:	
If $\mathbf{w}([\mathbf{S_i}]^{\mathbf{T}}.(T_{id\downarrow} s) \oplus \mathbf{r}) > n.\tau'$ **return**;	
Else	
$I_{i+1} = r$	
if $\mathbf{w}(s') \neq l$ **return**;	
Generate non-singular $\mathbf{[Q]} \in_R \mathbb{Z}_2^{l\times l}$	
$[\mathbf{S_{i+1}}] = [\mathbf{Q}].[\mathbf{S_i}] \in \mathbb{Z}_2^{l\times n}$	
where $rank(\mathbf{S_{i+1}}) = n$	
Compute $\mathbf{P_{i+1}} = [\mathbf{S_{i+1}}][\mathbf{S_{i+1}}]^+ \in \mathbb{Z}_2^{l\times l}$	
where $[\mathbf{S_{i+1}}]^+ = ([\mathbf{S_{i+1}}]^{\mathbf{T}}[\mathbf{S_{i+1}}])^{-\mathbf{1}}[\mathbf{S_{i+1}}]^{\mathbf{T}} \in \mathbb{Z}_2^{n\times l}$	
$\mathbf{P_i}' = [\mathbf{P_i}][\mathbf{Q}] \in \mathbb{Z}_2^{l\times l}$;	
$\mathbf{e}' \in_R \mathbf{Ber}_\tau^n$;	
$\mathbf{r}' := [\mathbf{S_i}]^{\mathbf{T}}.\ (T_{id\downarrow} s') \oplus \mathbf{e}'$	
$\xrightarrow{(\mathbf{P_i}', \mathbf{r}')}$	
	if $\mathbf{w}([\mathbf{S_i}]^{\mathbf{T}}.\ (T_{id\downarrow} s') \oplus \mathbf{r}') > n.\tau'$ **return**; else **accept**
	$\mathbf{S_{i+1}} = (\mathbf{P_i}'.\mathbf{S_i}) \in \mathbb{Z}_2^{l\times n}$
	if $rank([\mathbf{S_{i+1}}]) \neq n$ **return**;

Fig. 1. RFID Authentication Protocol

- Reader: Generate a random binary challenge string s, and sends it to a tag.
- Tag: Check the *hamming weight* of the string s and generate an n-bit noise vector $\mathbf{e}$, a random $2l$-bit challenge string s' for a reader with hamming weight l. Next an n-bit LPN problem is computed as $\mathbf{r} := [\mathbf{S_i}]^{\mathbf{T}} \cdot (T_{id\downarrow} s) \oplus \mathbf{e}$. To eliminate brute-force searching at the server end, maintain an index I_i and send it to the reader. Finally, update index I_{i+1} to r and send $(I_i, s', \mathbf{r})$ to the server.
- Reader: First search database to find a tuple $[I_i, T_{id}, S_{i-1}, S_i, P_{i-1}, P_i]$ with index I sent by the server. But searching might fail sometimes e.g., due to synchronization attack etc. If it fails, then apply brute-force method

targeting to explore $\mathbf{S_i}$ or $\mathbf{S_{i-1}}$ such that it satisfies LPN problem: $\mathbf{w}([\mathbf{S_i}]^{\mathbf{T}} \cdot (T_{id\downarrow}s) \oplus \mathbf{r}) \leq n \cdot \tau']$, or $[\mathbf{w}([\mathbf{S_{i-1}}]^{\mathbf{T}} \cdot (T_{id\downarrow}s) \oplus \mathbf{r}) \leq n \cdot \tau']$. If the brute-force method passes, it accepts the tag, update the index to I_{i+1} and enter *reader authentication* phase.

For *reader authentication*, it has secret S_i, P_i and other public parameters which has been derived from previous $(i-1)$ successful authentication sessions.

- Reader: First test whether *hamming weight* of s' is exactly l. Then generate a non singular binary matrix Q to update session key S_{i+1} as $[Q \cdot S_i]$ and compute pseudo inverse-matrix S^+_{i+1}, and P_{i+1} as $[S_{i+1} \cdot S^+_{i+1}]$. To send the new session key S_{i+1} to the tag and blinding the matrix Q, P'_i is computed by $[P_i \cdot Q]$ which is actually equivalent to a binary matrix $[S_i S_i^+ Q]$. Assume the adversary cannot reveal S_i from P'_i in polynomial time. Next, for reader authentication, generate an n-bit noise vector e' and compute multiple bit LPN problem as $\mathbf{r'} := [\mathbf{S_i}]^{\mathbf{T}} \cdot (T_{id\downarrow}s') \oplus \mathbf{e'}$. Finally answer the tag with string $(\mathbf{P_i}', \mathbf{r'})$.
- Tag: Check the *hamming weight* of $([\mathbf{S_i}]^{\mathbf{T}} \cdot (T_{id\downarrow}s') \oplus \mathbf{r'}) \leq n \cdot \tau'$ where $(n \cdot \tau')$ is the predefined accepted threshold value for the LPN problem. If this check passes, accept the reader and update session key $\mathbf{S_{i+1}}$ by $[(\mathbf{P_i}' \cdot \mathbf{S_i}) = (\mathbf{S_i S_i^+ Q} \cdot \mathbf{S_i}) = (\mathbf{S_i Q})]$ where $[\mathbf{S_i S_i^+ S_i} = \mathbf{S_i}]$[3]. However, if the check fails, tag's session key remains unchanged.

4 Security Analysis

4.1 SLPN Problem

We use the proof method similar to that described in [2] as Theorem 1. follows. Even though protocol in our model and that in [2] are different, a similar proof can be used as both are based on the $SLPN^*$ problem. Hardness of $SLPN^*$ can be defined using an indistinguishability game. More formally, security of the proof is based on the computational indistinguishability of the two oracles $SLPN^*$ and uniform distribution U_{2l}. From the protocol description it can be found that noise is a vector rather than a single bit; and the secret is not a vector but a pseudo-random matrix.

Theorem 1. *For any constant $\gamma > 0$, let $n = l/2 + \gamma$. If the $SLPN^*$ problem is (t, Q, ϵ)-hard then the authentication protocol from Figure 1. is (t', Q, ϵ')-secure against active adversaries, where the constants $(c_\gamma, c_\tau > 0)$ depend only on γ and τ respectively.*

$$t' = t - poly(Q, l) \quad \epsilon' = \epsilon + Q.2^{-c_\gamma \cdot l} + 2^{-c_\tau \cdot n} = \epsilon + 2^{-\theta(n)}$$

The protocol has completeness error $2^{-c_\tau \cdot n}$

[3] From the properties of pseudo-inverse matrix.

4.2 Man-in-the Middle Attack

The most sophisticated and realistic attack in RFID system is Man-in-the Middle (MIM) attack. Our protocol is MIM-secure against active attack from SLPN assumption. Note that, first the reader authenticates the tag and then vice versa. In case of tag authentication, it runs a two-round MIM-secure authentication protocol where *reader* chooses a random variable as challenge, and *tag* returns the response according to the challenge. Authentication tag $\gamma = (S, r : S^T f_k(s) \oplus e)$ where $f_k(s)$ is the secret key derivation function which uniquely encodes challenge s according to k by selecting l bits from the key[4] k. The main technical difficulty to build a secure MIM-free authentication from LPN is to make sure the secret key k does not leak from verification queries. In [2], they use randomize-mapping function $f_k(s) = (k \downarrow s : \mathbb{Z}_2^{2l} \to \mathbb{Z}_2^l)$ for some random s and proved that if LPN is hard then the construction is MIM-secure. We have twisted a little the original idea. In our construction, we remain both S and k secret which enhances security. We use EX-OR operation for hiding s' using T_{id} as key. Note that, XOR cipher is vulnerable to frequency analysis; therefore even if adversary compromises T_{id}, it cannot generate S_i for any subsequent sessions using only T_{id}. In the third phase of the protocol, we introduce pseudo-random matrix as blinding factor to transfer new session key S_{i+1} which is secure from pseudo-random matrix property assumption.

4.3 Pseudo-random Matrix

We followed security analysis in [13] where they claim that, having known the messages $XX^+Q \in \mathbb{Z}_2^{l \times l}$, it is impossible to recover the secrets $X \in \mathbb{Z}_2^{l \times n}$, or $Q \in \mathbb{Z}_2^{l \times l}$. Given $XX^+Q \in \mathbb{Z}_2^{l \times l}$, suppose that $rank(X) = r$, and

$$X^+X = \begin{pmatrix} I^{r \times r} & 0 \\ 0 & 0 \end{pmatrix} \Rightarrow X^+XQ = \begin{pmatrix} Q^{r \times r} & 0 \\ 0 & 0 \end{pmatrix}$$

where $I^{r \times r}$ is an *Identity matrix* and $Q^{r \times r}$ is the left upper sub-matrix of Q. Then the probability of an adversary to determine the correct Q is $2^{-(l-r)n}$. To ensure security we need to ensure $l \gg r$ which can be obtained by $l > n$. In our authentication protocol, we let $n \leq l/2$ to ensure large value of l.

4.4 Forward Security

For each operation, tag uses session key S_i and reader also use its corresponding P_i for verification of authentication tags. At the end of each valid session, (S_i, P_i) is updated with random matrix and previous key is deleted permanently in the tag. We say that even if S_i is exposed during authentication session i by the attacker, tag's privacy is fully guaranteed for $(i-1)$ period.

[4] We use T_{id} as the secret key k.

4.5 Backward Security

Typical RFID tags and their reader communicate only for a short period of time because of power constraint of a tag. Thus, either we restrict the adversary in a way that it can obtain neither T_{id} nor S_i at any time instance i, or there should exist some non-empty gap between the time of a reveal query and the attack, while tag is not accessible by the adversary. This entails the adversary miss the protocol transcripts needed to update the compromised secret key and hence our protocol claims **reduced** backward security.

4.6 Tracking a Tag

Protocol can resist tracking the tag due to the following reason: it refreshes the random vector (s, s', e, e'), updates the keys (P_i, S_i) while assumptions like SLPN problem, pseudo-random matrix makes the protocol indistinguishable from the adversarial perspective.

4.7 De-synchronization Attack

We introduce indexing of the tag to get rid of the attack. When the reader and the tag maintain synchronization, searching hash table becomes very fast with *direct match* technique. However, synchronization attack may take place in the third protocol transcript from the reader to the tag; while the tag may not receive (p', r') to update its shared key. In the later case, brute-force search will be used for successful authentication. Though it incurs worse performance, but after successful authentication synchronization would be recovered.

4.8 Tag Cloning

We use two different keys T_{id} and S_i for the tag. Therefore, even if the tag is cloned by malicious reader, we assume either of the keys is not compromised. For instance, an EPC generation 2 allows a password-enabled *secure state* configuration that prevents anyone from reading or writing onto a tag memory bank. Let T_{id} be stored in a password protected memory bank. Moreover, tag is not allowed to update it's key S_i until it authenticates the reader. This verification thwarts the cloning attack as well.

4.9 Replay Attack

Assuming that the random challenges sent by the reader and the tag are same in two different sessions, an adversary can launch *replay attack* by snooping the random numbers; but in our protocol, the reader queries the tag each time with a new random challenge s and then tag queries reader with random s', I_i. So, it is very unlikely to find a match between a pair of (I_i, t, r) from two different sessions of the same tag.

4.10 Privacy

First, we analyze our protocol using the *privacy model* in [15].

Theorem 2. *If the encryption in the protocol described in Fig. 1. is indistinguishable then the protocol is strong private for narrow adversaries.*

Proof: Given an adversary $\mathcal{A}$ that wins the privacy game with non-negligible advantage, we consider another adversary $\mathcal{B}$ that can break the *indistinguishaibility* game with non-negligible advantage described in *section* 4.1. The adversary $\mathcal{B}$ runs the adversary $\mathcal{A}$ to answer queries with the following exceptions:

- S, T_{id} are two different keys of the indistinguishaibility game.
- *SendTag* $(vtag, s)_b$: By retrieving the tag T_i and T_j references from the table D using virtual tag *vtag*; it generates two references $m_0 = \mathbf{w}([\mathbf{S_i}]^{\mathbf{T}}.(T_{i_{\downarrow}}s) \oplus \mathbf{r}) > n.\tau'$ and $m_1 = \mathbf{w}([\mathbf{S_j}]^{\mathbf{T}}.(T_{j_{\downarrow}}s) \oplus \mathbf{r}) > n.\tau'$. The references m_0, m_1 are sent to the indistinguishability oracle of SLPN problem which returns whether *hamming weight* satisfies $w \leq n.\tau'$ under one of the references .
- $\mathcal{B}$ cannot query for *Result*() oracle.

At the end of the game, $\mathcal{B}$ outputs according to $\mathcal{A}$'s guess. Hence, $\mathcal{B}$ is perfectly simulated for $\mathcal{A}$. If $\mathcal{A}$ breaks privacy then $\mathcal{B}$ wins indistinguishibility game; but indistinguishibility with only one call to the oracle is equivalent to indistinguishibility with multiple calls to the oracle that proves the *narrow privacy* of the protocol. □

In [16], authors have categorized RFID authentication protocols into *four* types according to their constructions and distinguished *eight* privacy levels by their natures on accessing *Corrupt*() oracle in the strategies of the adversary and whether *Result*() oracle is used or not.

- *Nil*: No privacy protection at all.
- *Weak*: Adversary has access to all oracles except *Corrupt* (T_i).
- *Forward* : Adversary has access to *Corrupt*(T_i) but other oracles are not allowed as *Corrupt*(T_i) oracles are accessed.
- *Destructive* : No restriction on accessing other oracles after *Corrupt* (T_i), but T_i is not allowed to use again.
- *Strong* : It is the strongest defined privacy level with no restrictions.

Each of these levels has its *narrow* counterpart to restrict the access of *Result*() oracle.

Our protocol belongs to *Type 2a* for construction where the shared key S_i has been updated just after the reader is authenticated. We now redefine our protocol privacy according to the model described in [16].

Without reader authentication, any adversary can keep querying a tag with any compatible reader until it is desynchronized with legitimate reader. Therefore, the tag's secret can only be desynchronized by one update. As the reader has both the keys S_i and S_{i-1}, in case of tag failure to update its shared key S_i, the reader can still try to authenticate the victim using the previous key S_{i-1} in

the next protocol conversation. Thus, it provides *weak* privacy to the protocol construction. Let an adversary $\mathcal{A}$ try to send authentication transcripts to the tag by blocking a valid reader authentication message, or by intercepting the tag in an online attack. This causes the tag to be in DoS attack or in a deadlock condition as it cannot update the key without reader authentication.

Theorem 3. *Protocol described in Fig. 1. is weak non-narrow privacy preserved.*

Due to lack of space, we remove the proof of the above theorem. That will appear in the full version. However, this narrow-forward privacy level attack can be reduced if tag accepts any value to update the key, but it is not practical; as in that case reader authentication message can be easily forged.

Table 2. Tag Resources and Security Comparison with HB family and Others

Scheme	**Storage**	**Computation** (major)	**Authentication**	**Security achieved**	**Hardware** (gates)
[2]	1 S	1 LPN	tag	1,4*	≈ 1600
HB [3]	1 S	1 LPN	tag		≈ 1600
HB$^+$ [6]	2 S	2 LPN	tag	7	≈ 1600
HB-MP[8]	2 S	1 LPN	tag	5,6,7,9	≈ 1600
HB-MP$^+$ [22]	2 S	1 LPN,1 HASH	tag	1,5,6,7,9	≈ 3500
F-HB [17]	1I , 1 S	1 PRNG,2 LPN	mutual	$1, 2, 4^*, 5, 6, 7, 9$	≈ 3500
ours	**1 I, 1 S**	**1 LPN,1 PIM**	**mutual**	1,2,3*,4,5,6,7,8,9	≈ **1600**
[21]	1 S	1 PRF,1 HASH	tag	2,4,6,8,9	≈ 6000
[18]	1I ,1 S	1 PRF	mutual	2,4*,6,8,9	≈ 6000
[19]	1 S	1 PRNG,1 UH	tag	2,4,9	≈ 3500
[20]	1 S	1 SC	mutual	2,4*,8,9	≈ 2000
[23]	1 S ,2 TS	1 HASH	tag	4*	≈ 1500
[24]	1 S , 1 TS, 1 RN	2 HASH	mutual	$4^*, 8, 9$	≈ 1500
[25]	1 RN,1 C, 1 TS,1 S	3 HASH	mutual	$2, 4^*, 6, 8, 9$	≈ 1500

where SC:= Stream Cipher; S:= Secret key; C:= Counter; I:= Index;
PRNG:= Psudo Random Number Generator; UH:= Universal Hash;
PIM:= Pseudo Inverse Matrix; LPN:= Learning parity from noise
TS:= Time Stamp; RN:= random number;
Security attributes: Man-in-the Middle attack(1), Forward Security (2), Backward Security (3), Reduced Backward Security (3*), High Privacy (4), Limited Privacy (4*) Tag tracking (5), De-synchronization (6), Tag Cloning (7), Replay attack (8), DoS (9).

5 Comparison and Performance Analysis

In case of tag, protocol operations include *two* random binary vector generation, *one* SLPN problem, *one* EX-OR operation, *three* binary linear matrix multiplication. For computation, we only consider SLPN problem and assume rest of

the operations (e.g., calculation hamming weight) be trivial in terms of computational complexity. The protocol is roughly as efficient as HB^+ protocol with just twice the key length. Since it is a reduction of LPN to SLPN problem, the protocol is secure against quantum adversaries assuming LPN is secure against such adversaries. There is a natural trade-off between communication cost and key size. For any constant c ($1 \leq c \leq n$), communication cost can be reduced by a factor of "c" by increasing the key size with the same factor.

Major computations of the proposed authentication scheme on the tag include linear binary matrix multiplication and LPN problem. And in case of storage, only a secret key and an index for the key. As bitwise XOR, matrix multiplication, hamming weight $w()$ and $(a_{\downarrow}b)$ are all binary operation, they can be easily implemented using bit-by-bit serialization to save hardware gates. In e-STREAM project, PRNG operation needs only 1,294 gates using Grain-v1[4]. The cost of a LPN problem and storing index and secret key may not be greater than PRNG and should be less than CRC. However, LPN problem can be implemented using a linear feedback shift register (LFSR) (for Transpose matrix), a 1-bit multiplier plus 1-bit accumulator (for binary multiplication), XOR gates (for $\oplus$ operation) 1-bit counter(for hamming weight) and a 1-bit comparator (for $a_{\downarrow}b$ operation). Thus, to achieve λ-bit security level; the overall hardware cost of the proposed protocol for the above mentioned functions on a tag is no more than 1600 gates including the cost of non-volatile memory to store secret key, index value and protocol intermediate values; and the protocol is suitable for Class-1 Generation-2 EPC tags where PRNG and CRC are used as hardware.

We demonstrate a comparative study on some general attributes e.g., storage consumption, major computations, authentication party, achieved security, approximate hardware cost etc. between our protocol and several HB-like and non-HB protocols in Table 2. It appears that although tag's hardware cost of the proposed protocol is optimal, yet it achieves most common security requirements. Additionally, it achieves $O(1)$ time complexity during synchronized state that resists brute-force searching in each authentication session. Alternatively, hardware cost of the reader is very expensive for the purpose of complex computing[5] resulting in reduced computing in tag and hence hardware. Besides that, hash-indexed searching technique at the reader where all the data related to a certain tag are stored efficiently as *index* reduces exhaustive database search at the reader end. As a consequence, in an RFID system with *remote authentication*[6], reader can use this index in *batch mode* operation to aggregate responses from several tags together to reduce communication cost between the reader and the server where each tag contains unique index within the reader's *field of view* at a specific time instance.

[5] Searching database, Generating pseudo-random matrix are the most complex part of the protocol.

[6] Tag readers are portable and server access is costly.

6 Conclusion

This paper presents a novel hardware-friendly RFID authentication protocol based on SLPN problem that can meet the hardware constraints of the EPC Class-1 generation-2 tags. In comparison to other protocols as described in Table 2, it requires less hardware and has achieved major security attributes. The protocol is also compliant to *strong private for narrow adversaries* privacy settings. Moreover, scalability of the protocol can be realized in synchronized mode and desynchronized mode that ensures infinite DoS resistance. Security and privacy can be protected as long as we allow adversary not to cope with both tag ID and the secret key simultaneously. In addition, the security and privacy proof follows standard model that uses indistinguishability as basic privacy notion. Our future research will focus on how to reduce the communication cost between the reader and server, assuming the wireless link between them is insecure, to figure a realistic privacy-preserving RFID environment.

References

1. Juels, A., Weis, S.A.: Authenticating Pervasive Devices with Human Protocols. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 293–308. Springer, Heidelberg (2005)
2. Kiltz, E., Pietrzak, K., Cash, D., Jain, A., Venturi, D.: Efficient Authentication from Hard Learning Problems. In: Paterson, K.G. (ed.) EUROCRYPT 2011. LNCS, vol. 6632, pp. 7–26. Springer, Heidelberg (2011)
3. Hopper, N.J., Blum, M.: Secure Human Identification Protocols. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 52–66. Springer, Heidelberg (2001)
4. Cid, C., Robshaw, M.: The eSTREAM Portfolio 2009 Annual Update (July 2009), `http://www.ecrypt.eu.org/stream/`
5. Katz, J., Shin, J.S., Smith, A.: Parallel and concurrent security of the HB and HB+ protocols. Journal of Cryptology 23(3), 402–421 (2010)
6. Gilbert, H., Robshaw, M., Sibert, H.: An active attack against HB+ - a provably secure lightweight authentication protocol. Cryptology ePrint Archive, Report 2005/237 (2005), `http://eprint.iacr.org/`
7. Bringer, J., Chabanne, H., Dottax, E.: HB++: a lightweight authentication protocol secure against some attacks. In: SecPerU, pp. 28–33 (2006)
8. Munilla, J., Peinado, A.: HB-MP: A further step in the HB-family of lightweight authentication protocols. Computer Networks 51(9), 2262–2267 (2007)
9. Gilbert, H., Robshaw, M.J.B., Seurin, Y.: Good Variants of HB^+ Are Hard to Find. In: Tsudik, G. (ed.) FC 2008. LNCS, vol. 5143, pp. 156–170. Springer, Heidelberg (2008)
10. Gilbert, H., Robshaw, M., Seurin, Y.: $HB^{\#}$: Increasing the Security and Efficiency of HB^+. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 361–378. Springer, Heidelberg (2008)
11. Ouafi, K., Overbeck, R., Vaudenay, S.: On the Security of $HB^{\#}$ against a Man-in-the-Middle Attack. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 108–124. Springer, Heidelberg (2008)
12. Moore, E.H.: On the reciprocal of the general algebraic matrix. Bulletin of the American Mathematical Society 26(9), 394–395 (1920), doi:10.1090/S0002-9904-1920-03322-7

13. Thuc, D.N., Hue, T.B.P., Van, H.D.: An Efficient Pseudo Inverse Matrix-Based Solution for Secure Auditing. In: IEEE-RIVF, pp. 7–12 (2010) ISBN: 978-1-4244-8072-2
14. Dodis, Y., Katz, J., Xu, S., Yung, M.: Key-Insulated Public Key Cryptosystems. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 65–82. Springer, Heidelberg (2002)
15. Hermans, J., Pashalidis, A., Vercauteren, F., Preneel, B.: A New RFID Privacy Model. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 568–587. Springer, Heidelberg (2011)
16. Ng, C.Y., Susilo, W., Mu, Y., Safavi-Naini, R.: New Privacy Results on Synchronized RFID Authentication Protocols against Tag Tracing. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 321–336. Springer, Heidelberg (2009)
17. Cao, X., O'Neill, M.: F-HB: An Efficient Forward Private Protocol. In: Workshop on Lightweight Security and Privacy: Devices, Protocols and Applications (Lightsec 2011), Istanbul, Turkey, March 14-15 (2011)
18. Le, T.V., Burmester, M., de Medeiros, B.: Universally Composable and Forward-secure RFID Authentication and Authenticated Key Exchange. In: ACM Symposium on InformAtion, Computer and Communications Security (ASIACCS) (March 2007)
19. Berbain, C., Billet, O., Etrog, J., Gilbert, H.: An Efficient Forward Private RFID Protocol. In: ACM Conference on Computer and Communications Security (CCS) (November 2009)
20. Billet, O., Etrog, J., Gilbert, H.: Lightweight Privacy Preserving Authentication for RFID Using a Stream Cipher. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 55–74. Springer, Heidelberg (2010)
21. Avoine, G., Oechslin, P.: A Scalable and Provably Secure Hash- Based RFID Protocol. In: IEEE International Workshop on Pervasive Computing and Communication Security (March 2005)
22. Leng, X., Mayes, K., Markantonakis, K.: HB-MP+ Protocol: An Improvement on the HB-MP Protocol. In: IEEE International Conference on RFID, pp. 118–124 (April 2008)
23. Tsudik, G.: Ya-trap: Yet another trivial rfid authentication protocol. In: PerCom Workshops, pp. 640–643 (2006)
24. Chatmon, C., van Le,T., Burmester, M.: Secure anonymous rfid authentication protocols. Computer & Information Sciences, Florida AM University, Tech. Rep. (2006)
25. He, L., Jin, S., Zhang, T., Li, N.: An enhanced 2-pass optimistic anonymous rfid authentication protocol with forward security. In: WiCOM, pp. 1–4 (2009)

Efficient Keyword Search over Encrypted Data with Fine-Grained Access Control in Hybrid Cloud

Jingwei Li[1], Jin Li[2], Xiaofeng Chen[3], Chunfu Jia[1], and Zheli Liu[1]

[1] College of Information Technical Science, Nankai University
{lijw,cfjia,liuzheli}@nankai.edu.cn
[2] School of Computer Science, Guangzhou University
lijin@gzhu.edu.cn
[3] State Key Laboratory of Integrated Service Networks, Xidian University
xfchen@xidian.edu.cn

Abstract. As cloud computing becomes prevalent, more and more sensitive information is being centralized into the cloud, which raises a new challenge on how to efficiently share the outsourced data in a fine-grained manner. Although searchable encryption allows for privacy-preserving keyword search over encrypted data in public cloud, it could not work effectively for supporting fine-grained access control over encrypted data simultaneously. In this paper, we consider to tackle the challenge above under a hybrid architecture in which a private cloud is introduced as an access interface between users and public cloud. We firstly propose a basic scheme allowing both exact keyword search and fine-grained access control over encrypted data. Furthermore, an advanced scheme supporting fuzzy keyword search is presented. In both schemes, overhead computation is securely outsourced to private cloud but only left behind the file encryption and decryption at user side. Finally, we demonstrate approaches to realize outsourcing cryptographic access control mechanism and further relieve the computational cost at user side.

1 Introduction

Cloud computing is capable of providing seemly unlimited "virtualized" resources to users as services across the whole Internet while hiding platform and implementation details [1]. Today's cloud service providers are able to offer both highly available storage and massively parallel computing resources at relatively low costs.

As cloud computing becomes prevalent, more and more sensitive information is being centralized into the cloud and securely shared by users with specified privileges. Due to the fact that data owners and cloud storage are no longer in the same trusted domain, it always follows the custom that sensitive data should be encrypted prior to outsourcing. In this case, the problem of searching becomes more challenge.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 490–502, 2012.

Searchable encryption initiated by Song et al. [20] for allowing privacy-preserving keyword search on encrypted data has been intensively researched in recent years [8][18][16][21][13][7][12]. Although can keep data both confidentiality and searchability for single user, considering on keyword search with fine-grained access control in multi-user setting – a more common scenario in cloud computing, existing searchable encryption may not work effectively. A naive approach to achieve this goal is through sharing the secret query key among the group of multiple users. Nevertheless, this not only increases the risk of key exposure, but also makes it hard to revoke user's searching ability. Curtmola et al. [6] proposed another approach based on the broadcast encryption technique, but such solution [6] only works in a broadcasting pattern (i.e. only one data owner and multiple users). Furthermore, broadcast encryption in general is a quite expensive primitive and the data owner may not execute it easily.

In this paper, aiming at efficiently solving the problem of fine-grained access control on encrypted data, we consider a hybrid architecture consisting of a public cloud and a private cloud. Therefore, users are able to utilize the private cloud as a proxy to securely outsource their data to the public cloud. Actually, this type of architecture is reasonable and has attracted more and more attention recently. For example, an enterprise might use a public cloud service, such as Amazon S3 for achieved data but continue to maintain in-house cloud for managing operational customer data. Under the hybrid architecture, we design a practical keyword search scheme which simultaneously supports fine-grained access control over encrypted data. In the proposed scheme, the operation of trapdoor generation is securely outsourced to the private cloud but left behind only the file encryption and decryption at user side. Beyond the exact keyword search, we present an advanced scheme under this new architecture to efficiently achieve fuzzy keyword search. With the help of the private cloud, the overhead computation of generating fuzzy keyword set and futile decryptions are both eliminated at user side. Finally, based on the observation in both of our schemes that the main computational cost at user side is the ABE scheme, we discuss the issue of outsourcing ABE to private cloud to further relieve the computational cost at user side.

1.1 Related Work

Symmetric Searchable Encryption. The symmetric searchable encryption (SSE) was proposed by Song et al. [20], in which a user stores his encrypted data in a semi-trusted server and later searches with a certain keyword. In [20], each keyword is independently encrypted under a specified two-layered encryption. Subsequently, Goh [9] introduced bloom filter to construct secure indexes for the keyword search, which allows server to check if a file contains a keyword without decrypting the entire file. A formal treatment to SSE was presented by Curtmola et al. [6]. They provided the security notions for SSE and presented "index" approach, in which an array and a look-up table are built for the entire file collection. Each entry of the array is used to store an encryption of file identifier set associated with a certain keyword, while the look-up table enables

one to locate and decrypt the appropriate element from array. Recently, aiming at providing SSE with efficient search and update, Liesdonk et al. [21] presented two schemes: the first one transforms each unique keyword to a searchable representation such that user can keep track of the set of associated keywords via appropriate trapdoor. The second one deployed a hash chain by applying repeatedly a hash function to an initial seed. Since only the user knows the seed, he/she is able to traverse the chain forward and backward, while the server is able to traverse the chain forward only.

Public-Key Searchable Encryption. Boneh et al. [2] firstly proposed and suggested a public-key encryption with keyword search (PEKS) construction. Such primitive can be widely applied in store-and-forward system, such as email system, in which a receiver can search for data that is encrypted under the receiver's public key. Subsequently, several improved constructions at different security levels have been presented [14][7]. Recently, many relevant extensions on keyword search have also been proposed, such as conjunctive keyword search [12][3], fuzzy keyword search [16], etc.

Twin Clouds Architecture. Recently, Bugiel et al. [4] provided an architecture consisting of twin clouds for secure outsourcing of data and arbitrary computations to an untrusted commodity cloud. Actually, their work is the inspiration of this paper, based on their twin clouds architecture, we consider to address the privacy-preserving keyword search problem simultaneously supporting fine-grained access control over encrypted data in public cloud. Moreover, the adversary model in this paper is weaker than that in [4]. Specifically, the private cloud in Bugiel et al.'s work [4] is required to be fully trusted, while it only needs to be semi-trusted in both of our schemes.

1.2 Organization

The rest of this paper is organized as follows. In Section 2, we propose the system model for keyword search with fine-grained access control in hybrid cloud. In Section 3, we propose an efficient keyword search scheme with security and efficiency analysis. An efficient fuzzy keyword search scheme is described in Section 4. In Section 5, we discuss the issue of how to further relieve the overhead computation at user side through outsourcing ABE. Finally we draw conclusion in Section 6.

2 System Model

2.1 Hybrid Architecture for Keyword Search

There are four entities defined in our system, that is, data owners/users, attribute authority, private cloud and public cloud. In this paper, each user is associated with attributes which can be transformed to a set of privileges owned by him/her. The attribute authority is responsible for issuing private keys to authorized users and the public cloud is to store the encrypted files of all the owners in a database

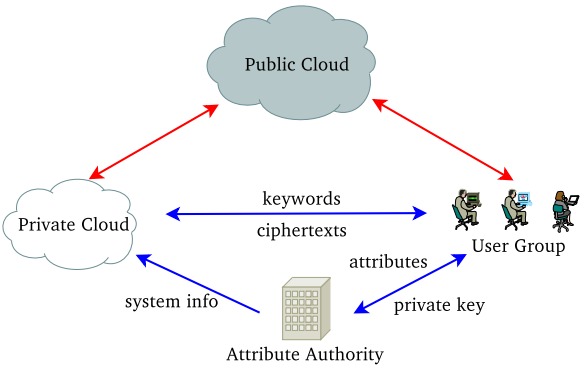


Fig. 1. Architecture for Keyword Search in Hybrid Cloud

and perform search for the users. More precisely, data owner uploads files along with an access policy and wants the files to be stored in the public cloud and shared with users whose attributes/privileges satisfy the required policy. The user can perform keyword search and decrypt the ciphertext retrieved from the public cloud. A private cloud is additionally introduced to facilitate user's secure usage of cloud service. Specifically, since the computing resources at user side are restricted and the public cloud is not fully trusted in practice, private cloud is able to provide users with an execution environment and infrastructure working as an interface between user and the public cloud. The interface offered by the private cloud allows user to securely submit files and queries to be securely stored and computed respectively.

2.2 Adversary Model

We assume that the public cloud and private cloud are both "honest-but-curious". Specifically they will follow our proposed protocol, but try to find out as much secret information as possible based on their possessions. Users would try to access data either within or out of the scopes of their privileges. Moreover, the communication channels involving the public cloud are assumed to be insecure. Therefore, two kinds of adversaries are considered in this system, that is, i) external adversaries, including the public cloud and the private cloud which aim to extract secret information as much as possible; ii) internal adversaries including revoked users and other unauthorized users who aim to obtain more privileges outside of their scopes.

Concerning on privacy, we require that all the files are sensitive and need to be fully protected against both public cloud and private cloud, while keywords are semi-sensitive and allowed to be known by the private cloud. Actually, approximately relaxing security demands by allowing keywords leakage to private cloud is innocuous because the private cloud in practice is able to be maintained by some organization itself.

3 Basic Scheme with Exact Keyword Search

3.1 Preliminary

Trapdoors of Keywords. Trapdoors of the keywords can be realized by applying a one-way function f which is defined as follows: given a keyword w and a private key k, the trapdoor of w can be defined as $T_{k,w} = f(k, w)$.

Attribute-Based Encryption. ABE was introduced by Sahai and Waters [19] and later formulized in [10] to construct fine-grained access control over encrypted data. Two flavors of ABE are classified, namely KP-ABE (key-policy ABE) and CP-ABE (ciphertext-policy ABE). In KP-ABE, attributes are used to describe the encrypted data and policies are built into users' keys, while it is inverse in CP-ABE. In this paper, we utilize the CP-ABE to facilitate key management and cryptographic access control in an expressive and efficient way. Let Ω and $\mathcal{AP}$ denote the attribute set and access policy. The CP-ABE scheme $\mathcal{ABE}$ consisting of four algorithms is described as follows:

- $\mathsf{Setup}_{\mathrm{ABE}}(1^\lambda)$: The setup algorithm takes as input – a security parameter 1^λ. It outputs the public key pk_{ABE} and the master key msk_{ABE}.
- $\mathsf{KeyGen}_{\mathrm{ABE}}(\Omega, msk_{\mathrm{ABE}})$: The key extraction algorithm takes as input – an attribute set Ω and the master key msk_{ABE}. It outputs the user's private key $sk_{\mathrm{ABE}}[\Omega]$.
- $\mathsf{Encrypt}_{\mathrm{ABE}}(\mathcal{AP}, m)$: The encryption algorithm takes as input – an access policy $\mathcal{AP}$ and a message m. It outputs the ciphertext ct.
- $\mathsf{Decrypt}_{\mathrm{ABE}}(ct, sk_{\mathrm{ABE}}[\Omega])$: The decryption algorithm takes as input – a ciphertext ct which was assumed to be encrypted under the access policy $\mathcal{AP}$ and the private key $sk_{\mathrm{ABE}}[\Omega]$. It outputs the message m if Ω satisfies the policy $\mathcal{AP}$ (denoted as $\mathcal{AP}(\Omega) = 1$), otherwise outputs the error symbol $\perp$.

3.2 Scheme Description

We now provide a description on how to deploy ABE to construct fine-grained access control system over searchable encrypted data.

System Setup. Initially, the attribute authority is to initialize attribute universe $\mathcal{U} = \{u_1,\ldots,u_n\}$ and privilege universe $\mathcal{PS} = \{p_1, \ldots, p_s\}$. In the system, these privileges are categorized via the users' attributes. Specifically, we can define a function $\Upsilon : 2^{\mathcal{U}} \rightarrow 2^{\mathcal{PS}}$ which maps any subset of $\mathcal{U}$ to some subset of $\mathcal{PS}$, then users with attributes Ω is assigned with privileges $\Upsilon(\Omega)$. The attribute authority randomly picks a symmetric key k_{p_i} for each $p_i \in \mathcal{PS}$, and sends the set of keys $\{k_{p_i}\}_{p_i \in \mathcal{PS}}$ to the private cloud.

Next, to initialize ABE primitive, the attribute authority chooses a security parameter 1^λ and runs the algorithm $\mathsf{Setup}_{\mathrm{ABE}}(1^\lambda)$ to obtain pk_{ABE} and msk_{ABE}. The public parameter is pk_{ABE}, and the master key is msk_{ABE}, which is kept secret by the attribute authority.

On the other hand, the private cloud maintains the set of symmetric keys $\{k_{p_i}\}_{p_i \in \mathcal{PS}}$ sent from the attribute authority and initializes a table $\mathcal{T}_{\text{user}}$ to record authorized users and their attributes.

New User Grant. Assuming that a new user identified by UID with attribute set Ω wants to join the system, he/she needs to make a request for a private key from the attribute authority. The authority assigns and returns a key $sk_{\text{ABE}}[\Omega]$ by running $\mathsf{KeyGen}_{\text{ABE}}(\Omega, msk_{\text{ABE}})$. The private cloud also stores UID and Ω into $\mathcal{T}_{\text{user}}$ as a newly authorized user's information.

File Uploading. Suppose that a data owner wants to share a file $\mathcal{F}$ identified by FID with users whose attributes satisfy an access policy $\mathcal{AP}$. Assume that the file $\mathcal{F}$ consists of a keyword set $\mathcal{W}$. Then, the owner randomly chooses a symmetric key k_{SE} from the key space and encrypts the file $\mathcal{F}$ with k_{SE} using standard symmetric key encryption algorithm such as AES to obtain encrypted file $ct_{\mathcal{F}}$. Subsequently, he/she runs the algorithm $\mathsf{Encrypt}_{\text{ABE}}(\mathcal{AP}, k_{\text{SE}})$ to obtain the ciphertext $ct_{k_{\text{SE}}}$ which is the encryption of the symmetric key k_{SE} with respect to the access policy $\mathcal{AP}$. The owner uploads $(\text{FID}, ct_{\mathcal{F}}, ct_{k_{\text{SE}}})$ to the public cloud. Furthermore, to generate the trapdoors for keywords in $\mathcal{W}$, the owner also sends $(\text{FID}, \mathcal{AP}, \mathcal{W})$ to the private cloud.

Upon receiving $(\text{FID}, \mathcal{AP}, \mathcal{W})$, the private cloud transforms the access policy $\mathcal{AP}$ into a set $\mathcal{PS}'$ of privileges. Then, for each $w_i \in \mathcal{W}$ and $p_i \in \mathcal{PS}'$, it computes $T_{p_i,w_i} = f(k_{p_i}, w_i)$. Finally, the private cloud sends $(\text{FID}, \{T_{p_i,w_i}\}_{p_i \in \mathcal{PS}', w_i \in \mathcal{W}})$ to public cloud as well.

File Retrieving. To search files containing a keyword w, the eligible user indentified by UID submits his/her query (UID, w) to the private cloud. The private cloud finds the entry (UID, Ω) in $\mathcal{T}_{\text{user}}$ and transforms the user's attributes to a set $\Upsilon(\Omega)$ of privileges. Then, for each $p_i \in \Upsilon(\Omega)$, it generates a trapdoor $T_{p_i,w} = f(k_{p_i}, w)$ and sends the collection $\{T_{p_i,w}\}_{p_i \in \Upsilon(\Omega)}$ to the public cloud.

Upon receiving the search request on $\{T_{p_i,w}\}_{p_i \in \Upsilon(\Omega)}$ from the private cloud, the public cloud finds all the matched trapdoors $T_{p_i,w}$ and returns all the $\{ct_{\mathcal{F}}, ct_{k_{\text{SE}}}\}$ corresponding to these matched trapdoors to the private cloud. The private cloud then forwards them to user who makes the search request. If the user has the privilege to access the encrypted file (determined by user's attribute set and the access policy specified in the ciphertext), he/she can successfully decrypt $ct_{k_{\text{SE}}}$ by running $\mathsf{Decrypt}_{\text{ABE}}(sk_{\text{ABE}}[\Omega], ct_{K_{\text{SE}}})$ to obtain k_{SE}. Then, he/she can utilize k_{SE} to decrypt and retrieve $\mathcal{F}$.

3.3 Improve Search Efficiency with Symbol-Based Trie

To enhance the search efficiency, a symbol-based trie is utilized to build an index stored in public cloud. More precisely, we can divide the output of one-way function f into l parts and predefine a set $\Delta = \{\alpha_1, \ldots, \alpha_t\}$ consisting of all the possible values in each part. Initially, the index based on symbol-based trie $\mathcal{I}$ has only a root node (denoted as $node_0$) which is consisted of $\emptyset$. Subsequently, it can be updated and searched as follows (an example of such tree can be shown in Fig. 2).

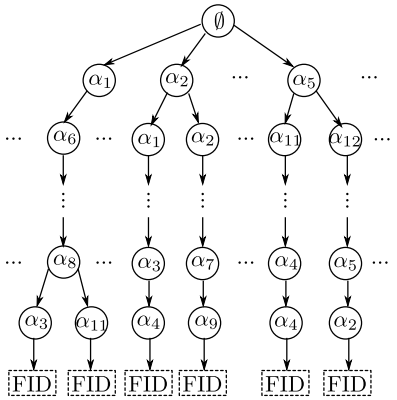


Fig. 2. An Example for Symbol-based Trie

- **Update.** Assume the data owner wants to outsource a file $\mathcal{F}$ identified by FID with keyword set $\mathcal{W}$, the public cloud will receive $(\text{FID}, ct_{\mathcal{F}}, ct_{k_{\text{SE}}})$ and $(\text{FID}, \{T_{p_i,w_i}\}_{p_i \in \mathcal{PS}', w_i \in \mathcal{W}})$ from data owner and private cloud respectively. Then, for each T_{p_i,w_i}, public cloud will add it into the trie index as the following steps.
  1) Public cloud parses T_{p_i,w_i} as a sequence of symbols $\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_l}$.
  2) Public cloud starts with the root node of trie: it scans all the children of the root node and checks whether there exists some child $node_1$ such that the symbol contained in $node_1$ equals to α_{i_1}. This action is performed in a top-down manner. In general, assuming that the subsequence of symbols $\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_{j-1}}$ has been matched and the current node is $node_{j-1}$, the public cloud will examine all the children of $node_{j-1}$ and attempt to find out some node $node_j$ such that the symbol contained in $node_j$ equals to α_{i_j}. If such node exists, current node is set as $node_j$ and $\alpha_{i_{j+1}}$ is the next matching object, otherwise jump to step 3).
  3) Assume that current node $node_j$ has no children to match the symbol $\alpha_{i_{j+1}}$, the public cloud will build nodes $node_{j+1}, \ldots, node_{j_l}$ for all the rest of symbols (i.e. $\alpha_{i_{j+1}}, \alpha_{i_{j+2}}, \ldots, \alpha_{i_l}$) respectively and link them as a node list appended with $node_j$. Finally, add another node identified by FID as the leafnode appended with $node_{j_l}$.
- **Search.** Assuming that the user wants to search outsourced files with keyword w and privileges $\Upsilon(\Omega)$, the public cloud will receive $\{T_{p_i,w}\}_{p_i \in \Upsilon(\Omega)}$ from private cloud. For each $T_{p_i,w}$, the public cloud will perform action similar to the three steps described above. One exception is that if matching is failed (i.e. current node has no children which can match the symbol), the search for $T_{p_i,w}$ is aborted. Otherwise, get the corressponding $(ct_{\mathcal{F}}, ct_{k_{\text{SE}}})$ through the identifier FID consisted in the leafnode.

3.4 Security Analysis

In the proposed keyword search scheme, the file is encrypted with a hybrid paradigm. Specifically, the symmetric key is utilized to encrypt file while it is encapsulated with ABE. By using a cryptographic strong cipher and secure ABE scheme, it is sufficient to assume that encrypted files leak zero information (except their respective lengths). Besides, the privacy-preserving query can be understood as a collection of l-length strings, the confidentiality/onewayness of which is guaranteed by the underlying trapdoor function. Moreover, concerning on the symbol-based trie, the public cloud is only able to perform prefix-mathching on keywords hidden by the one-way function. Therefore, nothing can be extracted except the access pattern or search pattern [6] (the search pattern in our setting is redefined as any information that can be derived from knowing whether two arbitrary searches are performed for the same keyword and privilege).

3.5 Efficiency Analysis

To upload a file, user has to run the symmetric key encryption and ABE encryption for a single time. Even if encrypting a file sized 100KB, it would not take more than 50ms using existing symmetric key encryption [22]. Thus, the main computational cost at user side is the ABE encryption which grows with the complexity of access policy. Furthermore, to generate searchable encrypted keywords with privileges, the private cloud is to perform the one-way function for $O(|\mathcal{PS}'| \times |\mathcal{W}|)$ times where $\mathcal{PS}'$ is the set of privileges transformed from the access policy and $\mathcal{W}$ is the keyword set associated with the file to be uploaded.

To make file retrieving, the private cloud has to generate trapdoors by running the one-way function for $|\Upsilon(\Omega)|$ times where Υ is defined as a transformation which maps an attribute set to some privileges and Ω is the user's attribute set. During search, the sybmbol-based trie in which the paths of trapdoors for different keywords (or same keyword but different privileges) are integrated by merging all the paths with the same prefix is utilized as the index. With such technique, it costs only $O(l)$ time for searching a single trapdoor. Furthermore, after receiving the results, user has to perfrom two-phased decryption including ABE decryption and symmetric key decryption.

In general, the main computational cost at user side is the ABE scheme. While existing ABE schemes are expensive (require a number of exponentiations and pairings during encryption and decryption respectively), we anticipate that any performance improvements in future schemes will directly result in similar performance gains for our scheme as well, since we use the ABE scheme in a black-box fashion.

3.6 Support User Revocation

Since the private cloud provides an access interface between user and public cloud, revocation is to be supported through rejecting the query request. Specifically, if attribute authority determines to revoke user with identity UID, it then

sends the revocation information to private cloud which just deletes the corresponding entry in $\mathcal{T}_{\text{user}}$. After that, such user's request is no longer responded by private cloud.

However, we point out that directly applying this will lead to cheating by user. Specifically, a revoked user is able to utilize other unrevoked user's UID to retrieve files and the private cloud cannot detect the cheating. Moreover, if a curious user intercepts the query submitted from other unrevoked user, he/she may present replay attack to obtain the response at any time. In order to solve such a problem, we utilize a message authentication code $\mathcal{MAC} = (\mathsf{KeyGen}_{\text{MAC}}, \mathsf{Mac}_{\text{MAC}}, \mathsf{Verify}_{\text{MAC}})$ to make a verification of user's identity, where $\mathsf{KeyGen}_{\text{MAC}}$ is the symmetric key generation algorithm, $\mathsf{Mac}_{\text{MAC}}$ is to use the symmetric key to generate an authentication code on message and $\mathsf{Verify}_{\text{MAC}}$ is the verifying algorithm to verify whether the code is valid on some message.

In the improved version, to respond to user's private request on attribute set Ω, authority needs to run $\mathsf{KeyGen}_{\text{MAC}}$ once to get a symmetric key $k_{\text{MAC}}[\Omega]$ and assign it with the user as a component of private key. Furthermore, $k_{\text{MAC}}[\Omega]$ is also to be sent to the private cloud as an item of the entry $(\text{UID}, \Omega, k_{\text{MAC}}[\Omega])$ added in $\mathcal{T}_{\text{user}}$. Then, when user makes file retrieving request, he/she has to run $\mathsf{Mac}_{\text{MAC}}(k_{\text{MAC}}, \text{UID}||w||time)$ to obtain the authentication code tag and send $(\text{UID}, w, tag, time)$ to private cloud, where $time$ is the present time. The private cloud fetches the corresponding entry $(\text{UID}, \Omega, k_{\text{MAC}}[\Omega])$ in $\mathcal{T}_{\text{user}}$ and makes a check firstly by running $\mathsf{Verify}_{\text{MAC}}(k_{\text{MAC}}, \text{UID}||w||time, tag)$. If the verification is failed, the cheating is detected.

4 Advanced Scheme with Fuzzy Keyword Search

In traditional fuzzy keyword search scheme [16], user has to compute trapdoors for all the relevant fuzzy keywords for both file uploading and retrieving, which leads to a large amount of overhead at user side. Additionally, when receiving the search results matching the fuzzy form of the query keyword, user has to pick his/her interests through decrypting all the ciphertexts. Typically, the files user desires to only occupy a small fraction of the returned results, thus many times of decryption at user side are less significant.

To fill the gap, we will show how to achieve efficient fuzzy keyword search under our architecture. Our basic idea is to outsource the heavy operation (i.e. trapdoor generation and futile decryption) to the private cloud and only left the light-weight computation (file encryption and decryption) at user side.

Before providing our solution, we need to define some notations. There are several methods to quantitatively measure the similarity of keywords. In this paper, we utilize the well-studied edit distance [15] for our purpose. Specifically, the edit distance $\mathsf{Ed}(w_1, w_2)$ between two words w_1 and w_2 is the number of operations (including substitution, deletion and insertion) required to transform either of them to the other. Given a keyword w, we can let $S_{w,d}$ denote the set of words w' satisfying $\mathsf{Ed}(w, w') \leq d$ for a certain integer d.

Suppose all the keywords used in this scheme are chosen from a finite word list $\mathcal{WL}$. For simplicity, the functionality of supporting user revocation is ommitted. Since the **System Setup** and **New User Grant** operate identically to that in Section 3.2, we only provide the other two phases as follows.

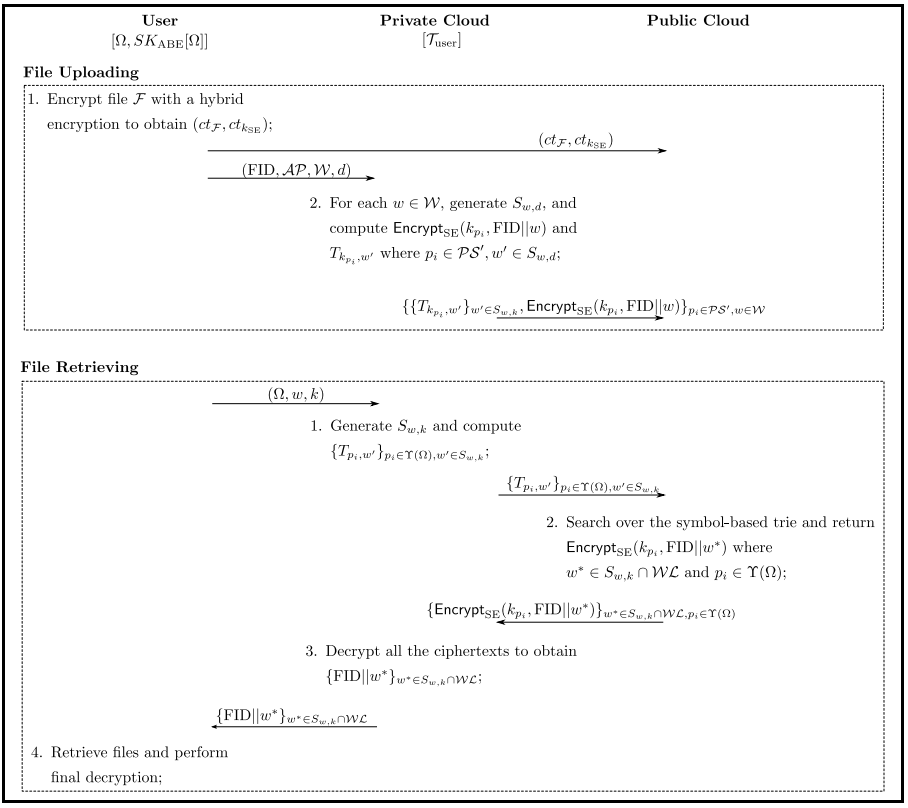


Fig. 3. File Uploading and Retrieving in Fuzzy Keyword Search

File Uploading. As shown in Fig. 3, when uploading a file $\mathcal{F}$, data owner performs a hybrid encryption on $\mathcal{F}$ by himself/herself but outsources the task of generating the fuzzy keyword set $\{S_{w,d}\}_{w \in \mathcal{W}}$ and further generating trapdoor for each $w' \in S_{w,d}$ to the private cloud where $\mathcal{W}$ is the keyword set associated with $\mathcal{F}$.

File Retrieving. As shown in Fig. 3, when retrieving file, the private cloud works as a proxy. Specifically, it firstly translates user's query into a set of trapdoors. Later, upon receiving the search results returned by public cloud, the private cloud is to perform decryption on them to make user's retrieving could be straightforwardly imposed on plaintext.

We state that with the help of the private cloud, fuzzy keyword search can be presented effciently. More precisely, user only needs to execute (hybrid) encryption and decryption but outsources the overhead computations including fuzzy keyword set generation and trapdoor generation to private cloud. Moreover, since user can straightforwardly perform screening on plaintext $\{\mathrm{FID}||w^*\}_{w^* \in S_{w,k} \cap \mathcal{WL}}$ and fetch corresponding encrypted files from public cloud, the futile decryptions are eliminated as well.

5 Discussion

As the efficiency analysis in Section 3.5, the main computational cost at user side is the ABE scheme (actually, the same conclusion can be drawn in our advanced scheme as well). Though the performance of existing ABE schemes is not satisfactory, we can outsource some expensive operations to the private cloud to relieve the computational overhead at user side. Actually, such paradigm has been investigated in some recent work [11][23][17][5] as well.

To outsource decryption, after authorization, user with attributes Ω can compute his/her blinded private key $\widetilde{SK}_{\mathrm{ABE}}[\Omega]$ with a randomly picked blinding factor t, and deliver $\widetilde{SK}_{\mathrm{ABE}}[\Omega]$ to the private cloud to be stored. In this paradigm, the private cloud will work as a proxy during decryption. More precisely, after receiving the search results sent from public cloud, the private cloud performs a partial decryption with $\widetilde{SK}_{\mathrm{ABE}}[\Omega]$ and forwards to user the partially decrypted ciphertext. Finally, user decrypts it using his/her original private key (the detail for outsourcing decryption can be refered in [23][11]). With this paradigm, the computational cost during decryption at user side can be reduced to constant (nearly a few exponentiations).

Beyond outsourcing decryption, we are able to outsource most of the computational cost during encryption as well. More precisely, a trival policy $\mathcal{AP}_\theta$ will be introduced and the ABE encryption is performed with a hybrid policy $\mathcal{AP}_\theta \wedge \mathcal{AP}$ where $\wedge$ is an AND gate connecting two sub-policies. The reason that we say it is trival is that it will not affect the global access control in the system. To achieve this, we can append some default attributes with each request attribute set such that $\mathcal{AP}_\theta$ is able to be satisfied by any user. Then, the secret s which is used to blind message during ABE encryption can be split into two parts s_1 and s_2. User utilizes s_1 to perform encryption with $\mathcal{AP}_\theta$ while $\mathcal{AP}$ is to be encrypted with by the private cloud using s_2 (a detail for outsourcing encryption can be refered in [17]). With this paradigm, the computational cost during encryption at user side can be reduced to constant (nearly a few exponentiations).

6 Conclusion

In this paper, aiming at efficiently solving the problem of fine-grained access control on searchable encrypted data, we consider a hybrid architecture in which a private cloud is introduced as an access interface between user and the public cloud. Under the hybrid architecture, we design a practical keyword search

scheme which simultaneously supports fine-grained access control over encrypted data. Beyond the exact keyword search, we present an advanced scheme under this new architecture to efficiently achieve fuzzy keyword search. Finally, based on the observation in both of our schemes that the main computational cost at user side is the ABE scheme, we discuss the issue of outsourcing ABE to private cloud to further relieve the computational cost at user side.

Acknowledgements. This work is supported by the National Natural Science Foundation of China (Nos. 61272423, 60973141, 61100224, 60970144 and 61272455), Fundamental Research Funds for the Central Universities, Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20100031110030), Funds of Key Lab of Fujian Province University Network Security and Cryptology (No. 2011004), Natural Science Foundation of Guangdong Province (No. 10451009101004573), and Foundation for Distinguished Young Talents in Higher Education of Guangdong Province (No. LYM10106).

References

1. Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., Zaharia, M.: A view of cloud computing. Commun. ACM 53(4), 50–58 (2010)
2. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
3. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
4. Bugiel, S., Nürnberger, S., Sadeghi, A.R., Schneider, T.: Twin clouds: An architecture for secure cloud computing. In: Workshop on Cryptography and Security in Clouds, WCSC 2011 (2011)
5. Chen, X., Li, J., Ma, J., Tang, Q., Lou, W.: New Algorithms for Secure Outsourcing of Modular Exponentiations. In: Foresti, S., Yung, M., Martinelli, F. (eds.) ESORICS 2012. LNCS, vol. 7459, pp. 541–556. Springer, Heidelberg (2012)
6. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable symmetric encryption: improved definitions and efficient constructions. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, pp. 79–88. ACM, New York (2006)
7. Di Crescenzo, G., Saraswat, V.: Public Key Encryption with Searchable Keywords Based on Jacobi Symbols. In: Srinathan, K., Pandu Rangan, C., Yung, M. (eds.) INDOCRYPT 2007. LNCS, vol. 4859, pp. 282–296. Springer, Heidelberg (2007)
8. Dong, C., Russello, G., Dulay, N.: Shared and searchable encrypted data for untrusted servers. Journal of Computer Security 19(3), 367–397 (2011)
9. Goh, E.J.: Secure indexes. An early version of this paper first appeared on the Cryptology ePrint Archive (October 2003)
10. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, pp. 89–98 (2006)

11. Green, M., Hohenberger, S., Waters, B.: Outsourcing the decryption of abe ciphertexts. In: Proceedings of the 20th USENIX Conference on Security, SEC 2011, pp. 34–34. USENIX Association, Berkeley (2011)
12. Hwang, Y.H., Lee, P.J.: Public Key Encryption with Conjunctive Keyword Search and Its Extension to a Multi-user System. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 2–22. Springer, Heidelberg (2007)
13. Ji, S., Li, G., Li, C., Feng, J.: Efficient interactive fuzzy keyword search. In: Proceedings of the 18th International Conference on World Wide Web, WWW 2009, pp. 371–380. ACM, New York (2009)
14. Khader, D.: Public Key Encryption with Keyword Search Based on K-Resilient IBE. In: Gavrilova, M., Gervasi, O., Kumar, V., Kenneth Tan, C.J., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3982, pp. 298–308. Springer, Heidelberg (2006)
15. Levenshtein, V.: Binary codes capable of correcting spurious insertions and deletions of ones. Problems of Information Transmission 1, 8–17 (1965)
16. Li, J., Wang, Q., Wang, C., Cao, N., Ren, K., Lou, W.: Fuzzy keyword search over encrypted data in cloud computing. In: Proceedings IEEE INFOCOM, pp. 1–5 (March 2010)
17. Li, J., Jia, C., Li, J., Chen, X.: Outsourcing encryption of attribute-based encryption with mapreduce. In: 14th International Conference on Information and Communications Security, ICICS (2012)
18. Li, M., Yu, S., Cao, N., Lou, W.: Authorized private keyword search over encrypted data in cloud computing. In: 2011 31st International Conference on Distributed Computing Systems (ICDCS), pp. 383–392 (June 2011)
19. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
20. Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: IEEE Symposium on Security and Privacy, pp. 44–55 (2000)
21. van Liesdonk, P., Sedghi, S., Doumen, J., Hartel, P., Jonker, W.: Computationally Efficient Searchable Symmetric Encryption. In: Jonker, W., Petković, M. (eds.) SDM 2010. LNCS, vol. 6358, pp. 87–100. Springer, Heidelberg (2010)
22. Weerasinghe, T.: Secrecy and performance analysis of symmetric key encryption algorithms. International Journal of Information & Network Security (IJINS) 1(2), 77–87 (2012)
23. Zhou, Z., Huang, D.: Efficient and secure data storage operations for mobile cloud computing. Cryptology ePrint Archive, Report 2011/185 (2011)

Masque: Access Control for Interactive Sharing of Encrypted Data in Social Networks*

Huimin Shuai[1,2] and Wen Tao Zhu[1]

[1] State Key Laboratory of Information Security
Institute of Information Engineering, Chinese Academy of Sciences
87C Min Zhuang Road, Beijing 100193, China
[2] Graduate University of Chinese Academy of Sciences
19A Yu Quan Road, Beijing 100049, China
hmshuai@is.ac.cn, wtzhu@ieee.org

Abstract. Making friends by sharing personal data has become popular in online social networks (OSNs). Security is a major concern, as an OSN service provider (OSN-SP) is semi-trusted and for-profit, while malicious users might also gather data for improper uses. Encryption of data is a straightforward solution, but interactive sharing of encrypted data becomes a challenging task. In this paper, we propose Masque, a novel access control mechanism employing attribute-based encryption (ABE), as a hierarchical solution for interactive sharing of encrypted data in OSNs. Based on key-policy ABE, it allows the OSN-SP to manage users at a high level but without being able to access their sensitive data. At the same time, based on ciphertext-policy ABE, it enables users to customize their own access policies specifically.

Keywords: Online social network, security, attribute-based encryption, data sharing.

1 Introduction

Online social network (OSN) applications such as making friends by sharing personal data like photos or videos within a group have become immensely popular. For example, Alice might want to share her photos in Tibet to make new friends in her college who are also interested in traveling in Tibet. The primary motivation of such applications is to provide a network for users to find potential friends with similar interests. However, security and sharing are a pair of paradoxical requirements. Currently, in most of such applications, the OSN service provider (OSN-SP) is assumed to be fully trusted. That is, users have to entrust their sensitive data to the OSN-SP which is the only one that is responsible for data security and privacy. In the real world, however, the OSN-SP is usually a semi-trusted for-profit server, which will honestly follow the designated protocol

* This work was supported by the National Natural Science Foundation of China under Grants 60970138 and 61272479, and also by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA06010702.

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 503–515, 2012.

but might collect users' private information and sell them to other organizations such as advertising companies. Even if we assume that the OSN-SP is fully trusted, access policies in such applications are often coarse-grained and inflexible. Typically, a user, say Alice, can only choose to either publish her personal data to all users or grant the access merely to her approved friends by manually maintaining an access control list. In the former case, no access policy is specified and sensitive data remain unsheltered in the OSN. In the latter one, only approved friends can view Alice's data, which contradicts the motivation of such applications to share data and make new friends.

So, the access control schemes used in current OSNs only provide basic and coarse access control. Requirements of data security and privacy are motivating the needs to encrypt sensitive data before releasing them. Besides, hierarchical access policy is preferable, where the OSN-SP is capable of specifying basic access policies for each social group while at the same time users could further customize their own access policies based on the basic one. Then, users could share their encrypted data by specifying the credentials of recipients instead of maintaining a list of approved friends.

In this paper, we present Masque, an access control mechanism based on attribute-based encryption (ABE). Masque supports hierarchical fine-grained access control for interactive sharing of encrypted data in OSNs, where the OSN-SP could manage users at a high level and specify basic access policies for social groups without actual access of sensitive user data. A registered user, say Alice, could establish social groups to potentially help her friends to make new friends without compromise of data security and privacy. Each member of Alice's group could customize fine-grained access policies based on the group's basic access policy and share personal data with the ones who have similar interests. We will show that Masque is proven secure.

The rest of the paper is organized as follows. Section 2 is related work. Preliminaries are given in Section 3. Section 4 presents the construction of Masque. Proof of security is presented in Section 5. Section 6 concludes this paper.

2 Related Work

2.1 Attribute-Based Encryption (ABE)

Inspired by previous work on identity-based encryption [1] [2], Sahai and Waters presented a vision in [3] that the data owner provides a predicate $f(\cdot)$ specifying how he wants to share his data, and each user is ascribed a secret key associated with his credential X; a user with credential X can decrypt a ciphertext encrypted with predicate $f(\cdot)$ if and only if $f(X) = 1$. A particular formulation, called attribute-based encryption (ABE) was also presented in [3], where a user's credential X is represented by a set of predefined attributes and the predicate is realized as an access policy represented by a logical formula over a set of attributes. Later, [4] and [5] clarified the concept of ABE and two complementary forms of ABE were presented, respectively key-policy ABE (KP-ABE) in [4] and ciphertext-policy ABE (CP-ABE) in [5]. In KP-ABE, predefined attributes

are used to annotate the ciphertexts and logical formulae over these attributes are embedded into users' secret keys. CP-ABE is complementary in that the predefined attributes are used to describe the users' secret keys and the logical formulas are embedded into the ciphertext. ABE is an ideal solution to achieving fine-grained expressive access control, as it allows an encryptor to specify recipients by customizing an access predicate $f(\cdot)$ in terms of logical formula over attributes instead of maintaining an access control list. In this paper, following the spirit of both KP-ABE and CP-ABE, we integrate them into a single ABE scheme, where the high-level access policies are KP-based and the specific access policies are CP-based.

2.2 ABE in Online Social Networks

ABE has been widely used in recent efforts concerning access control in OSNs, such as in [6], [8] and [9]. However, the direct adoption of the primitive algorithms has to assume an OSN-SP to be fully trusted, while in real scenarios, the OSN-SP is usually a semi-trusted for-profit server and should not access users' sensitive data. It is preferable that the OSN-SP has certain degree of authority to manage the users in the OSNs by specifying basic access policies for different social groups. Then any member of a social group can further customize her own access policy specifically to share personal data only with qualified users.

The above necessitates a hierarchical fine-grained access control mechanism for interactive sharing of encrypted data in OSNs, which has not been tackled in the literature.

2.3 Data Security and Privacy in Online Social Networks

An OSN is a multi-user service much like a jungle of good and evil. Users from different backgrounds could register in the OSN to get the service. Data security and privacy are crucial despite the needs of sharing information to make friends.

ABE is a good solution to access control in OSNs. The main challenge is to prevent snoops from the OSN-SP or attacks from colluding adversaries (i.e., ineligible users who conspire). In this paper, we present a novel ABE-based hierarchical fine-grained access control mechanism for interactive sharing of encrypted data in OSNs, which achieves provable security.

3 Preliminaries

3.1 Bilinear Map

Let G_1, G_2 be two multiplicative cyclic groups of prime order p. Let g be a generator of G_1 and e be a bilinear map $e : G_1 \times G_1 \to G_2$. The bilinear map e has the following properties:

1. Bilinearity: For all $g_1, g_2 \in G_1, a, b \in \mathbb{Z}_p$, we have $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
2. Non-degeneracy: $e(g, g) \neq 1$.

3.2 Access Policy

Following ABE, in our scheme the access policy is a logical formula of a set of predefined attributes $\mathcal{A}$. Further, the access policy will be represented by an access tree T, where each non-leaf node x is a logic gate (AND/OR gate) and the function $child(x)$ returns the number of child nodes of x. Each child of a non-leaf node x will be assigned an index from 1 to $child(x)$. $parent(x)$ returns the parent node of x and $index(x)$ returns the index of x ordered by his parent node. Each leaf node x is associated with one of the attributes $att(x)$ in $\mathcal{A}$.

In this paper, there are two kinds of access trees respectively for a social group and for a specific user. Suppose Alice builds a social group and we denote the access tree for her group as T_{AG}. A specific user, say Danny, could customize his own access tree based on T_{AG} and we denote Danny's self-customized access tree as T_D.

4 Construction of Masque

4.1 Overview of Masque

There are basically two kinds of entities involved in Masque: the OSN service provider (OSN-SP) and users. The OSN-SP is a semi-trusted for-profit server which will follow the designated protocol but might furtively collect and sell users' sensitive information to other organizations. Users must first register on the OSN-SP side to get their basic credentials described by a set of attributes. A group leader is a special registered user who is authorized to create a social group with a basic access policy specified by the OSN-SP. The group leader's friends can join the group and each of them will be assigned a specific credential in the group based on his basic credential. Each group member could make new friends with each other by sharing data concerning similar subjects, just like in a masquerade. Each member will encrypt sensitive data by embedding a self-defined access policy into the ciphertext. Note that the self-defined access policy must comply with the group's basic access policy specified by the OSN-SP and here, the OSN-SP cannot access any of the users' sensitive data. Other members of the group could decrypt the ciphertext if and only if their secret keys match the embedded access policy.

4.2 Construction of Algorithms

Masque consists of seven algorithms: **Prepare**, **Register**, **ApplyToBuildGroup**, **BuildGroup**, **KeyGeneration**, **Encrypt**, and **Decrypt**. They are specified as follows.

Prepare. The OSN-SP initiates and prepares for the entire service.

1. The OSN-SP generates two multiplicative cyclic groups of prime order p: G_1 and G_2. Let g be a generator of G_1 and e be a bilinear map $e : G_1 \times G_1 \rightarrow G_2$.

2. The OSN-SP specifies the basic attribute set $\mathcal{B}$ and randomly chooses a secret value for each $b_j \in \mathcal{B}$ denoted as:

$$b_j \in \mathcal{B} : t_j \in \mathbb{Z}_p^*,\ b_j \leftrightarrow t_j.$$

Here, to facilitate expression, let:

$$\mathcal{B} = (\text{“Gender”},\ \text{“Age”},\ \text{“City”},\ \text{“Occupation”},\ \text{“Religion” ...}).$$

3. The OSN-SP randomly chooses two secrete values $\alpha, \beta \in \mathbb{Z}_p^*$. Public key for the Entire service PK_E are constructed as follows:

$$PK_E = (G_1, G_2, p, e, g, \mathcal{B}, g^{\frac{\alpha}{\beta}}).$$

Master key for the Entire service MK_E is secretly held by the OSN-SP and is constructed as:

$$MK_E = (\alpha, \beta, \{b_j \in \mathcal{B} : t_j\}).$$

Register. Each user, say Alice, should first register on the OSN-SP side to get the service. The OSN-SP first assigns a set of attributes $\mathcal{B}_A \subseteq \mathcal{B}$ to Alice to describe her credential, for example:

$\mathcal{B}_A$ = (“Gender=Female”, “Age=28”, “City=Boston, Seattle”, “Occupation=Editor”).

Subsequently, the OSN-SP randomly chooses a unique value $I_A \in \mathbb{Z}_p^*$ for Alice and issues a receipt R_A for Alice's registration as follows:

$$R_A = (g^{\frac{I_A}{\beta}}, \{b_j \in \mathcal{B}_A, b_j \leftrightarrow t_j : g^{t_j}\}).$$

R_A will be secretly held by Alice.

ApplyToBuildGroup. Any registered user can apply on the OSN-SP side to build a social group. Suppose Alice applies to build a social group to share photos with the access policy:

“Occupation=Editor/Journalist AND (City=Boston OR Age<30)”.

Once the access policy is approved by the OSN-SP, the OSN-SP will specify a basic access tree T_{AG} for Alice's Group accordingly as shown in Fig. 1. Note that Alice can only process the attributes in $\mathcal{B}_A$ assigned by the OSN-SP. Then, the OSN-SP associates a polynomial P_x to each node $x \in T_{AG}$ in a top-down manner.

1. Starting from the root node r, set the degree of P_r as $d_r = k_r - 1$ and $P_r(0) = v$, where $v \in \mathbb{Z}_p^*$ is randomly and uniquely chosen by the OSN-SP. Then, the OSN-SP randomly finds d_r other points of P_r to define P_r completely.
2. For any other non-leaf node $x \neq r$, set $P_x(0) = P_{parent(x)}(index(x))$. Then, the OSN-SP randomly chooses d_x other points of P_x to define P_x completely.
3. The OSN-SP associates each leaf node x with an attribute $att(x) \in \mathcal{B}_A$ and the attribute set associated with T_{AG} is denoted as $\mathcal{B}_G \subseteq \mathcal{B}_A$. Here, $\mathcal{B}_G$=(“Age<30”, “City=Boston”, “Occupation=Editor/Journalist”).

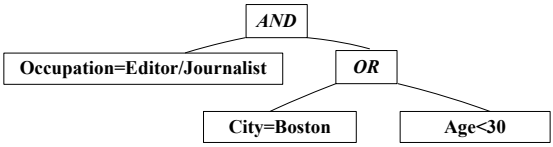


Fig. 1. Basic access tree for Alice's Group: T_{AG}

Finally, the OSN-SP generates a receipt R_{AG} for Alice to build her social group:

$$R_{AG} = (T_{AG}, g^{\beta v}, g^{vI_A}, \{x \in T_{AG}, att(x) \leftrightarrow t_x : g^{\frac{I_A P_x(0)}{t_x}}\}).$$

R_{AG} will be secretly held by Alice.

BuildGroup. Within Alice's group, she could define some new attributes called self-defined attributes in order to further describe the detailed subjects of the photos. Suppose the self-defined attribute set by Alice is:

$$\mathcal{A} = (\text{“Travel”, “Gourmet Food”, “Cars”, “Cosmetics”}).$$

Hence, the final attribute set for Alice's group is $\mathcal{A}_G = \mathcal{B}_G \cup \mathcal{A}$. For each $a_i \in \mathcal{A}_G$, Alice randomly picks $s_i \in \mathbb{Z}_p^*, s_i \leftrightarrow a_i$ and then, Alice creates the group public key PK_{AG} and group master key MK_{AG} with R_A and R_{AG} as:

$$PK_{AG} = (T_{AG}, g^{\beta v}),$$

$$MK_{AG} = (R_A, g^{vI_A}, \{x \in T_{AG}, att(x) \leftrightarrow t_x : g^{\frac{I_A P_x(0)}{t_x}}\}, \{a_i \in \mathcal{A}_G : s_j\}).$$

KeyGeneration. Alice's friends, say Danny, can apply to join Alice's group to share photos concerning certain subjects. Suppose the attribute set associated with Danny's credential is:

$$\mathcal{B}_D = (\text{“Gender=Male”, “Age=26”, “City=Boston”, “Occupation=Journalist”}).$$

Danny applies for some attributes in $\mathcal{A}$ according to his own interests, such as “Travel” and “Gourmet Food”. After being approved by Alice, Danny will be assigned a set of new attributes $\mathcal{A}_D \subseteq \mathcal{A}, \mathcal{A}_D$=(“Travel”, “Gourmet Food”) and the final attribute set of Danny in Alice's group is $\mathcal{D} = \mathcal{B}_D \cup \mathcal{A}_D$. Subsequently, Alice randomly picks $L_D \in \mathbb{Z}_p^*$ for Danny and computes with PK_E and R_A:

$$g^{\frac{\alpha}{\beta}} \cdot (g^{\frac{I_A}{\beta}})^{L_D} = g^{\frac{\alpha + I_A L_D}{\beta}}.$$

For each leaf node $x \in T_{AG}$, if $att(x) \in \mathcal{B}_D$, $att(x) \leftrightarrow t_x$, Alice computes with MK_{AG} as follows:

$$(g^{\frac{I_A P_x(0)}{t_x}})^{L_D} = g^{\frac{I_A L_D P_x(0)}{t_x}}.$$

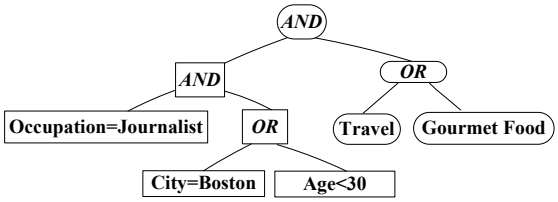


Fig. 2. Self-customized access tree by Danny in Alice's group: T_D. The value of attribute "Occupation" has been restricted as "Journalist" only.

For each attribute $a_j \in \mathcal{A}_D \cup (\mathcal{B}_D \cap \mathcal{B}_G)$, Alice computes with MK_{AG} as follows:

$$(g^{vI_A})^{\frac{L_D}{s_j}} = g^{\frac{vI_A L_D}{s_j}} \text{ and } g^{s_j}.$$

Finally, Alice grants the secret key SK_D to Danny. SK_D will be used to decrypt encrypted photos posted by other members of the group:

$$SK_D = (g^{\frac{\alpha + I_A L_D}{\beta}}, \{x \in T_{AG}, att(x) \in \mathcal{B}_D, att(x) \leftrightarrow t_x : g^{\frac{I_A L_D P_x(0)}{t_x}}\},$$

$$\{a_j \in \mathcal{A}_D \cup (\mathcal{B}_D \cap \mathcal{B}_G) : g^{\frac{vI_A L_D}{s_j}}, g^{s_j}\}).$$

Encrypt. Each group member, say Danny, can customize his own access tree T_D by adding new nodes into T_{AG} or restricting the original values of the nodes in T_{AG} tighter, such as restricting "Occupation" from "Editor/Journalist" to "Journalist" as shown in Fig. 2, where the nodes in ellipses are added by Danny. The corresponding access policy of T_D is:

"Occupation=Journalist AND (City=Boston OR Age<30) AND (Travel OR Gourmet Food)".

Danny can encrypt his photos by embedding T_D to the ciphertext and only members whose secret keys satisfy the embedded access policy T_D can decrypt the ciphertext.

Danny constructs T_D as follows. First, Danny associates a random value n_x to each node $x \in T_D$ in a top-down manner. Starting from the root node r, Danny randomly picks $n \in \mathbb{Z}_p^*$. Mark all other nodes as unassigned except the root node r. Recursively, for each assigned non-leaf node x, suppose its value is n_x, Danny assigns values to x's child as follows:

1. AND Gate. If x is an AND gate and its children are unassigned, Danny randomly and uniquely sets the value of each child y as $n_y \in \mathbb{Z}_P^*$ except the last one as:

$$n_{child(x)} = (n_x - \sum_{y=1}^{child(x)-1} n_y) \bmod p.$$

   Mark node x as assigned.

2. OR Gate. If x is an OR gate and its children y are unassigned, Danny sets $n_y = n_x$. Mark x as assigned.

Each leaf node $x \in T_D$ is associated with an attribute $att(x) \in \mathcal{A}_G$. Subsequently, Danny randomly chooses $q \in \mathbb{Z}_p^*$ and computes with R_D as:

$$D_1 = (x \in T_D, att(x) \in \mathcal{B}_D, att(x) \leftrightarrow t_x : C_x = g^{t_x q}).$$

Then, Danny computes with SK_D as:

$$D_2 = (x \in T_D, att(x) \in \mathcal{A}_D \cup (\mathcal{B}_D \cap \mathcal{B}_G), att(x) \leftrightarrow s_x : C'_x = g^{s_x n_x}).$$

Finally, Danny computes with PK_E and PK_{AG} as follows:

$$D_3 = g^{\beta v(n+q)},\ D_4 = e(g^{\frac{\alpha}{\beta}}, g^{\beta v})^{n+q} = e(g,g)^{\alpha v(n+q)}.$$

and encrypts his photo M as:

$$C = (T_D, D_0 = MD_4, D_1, D_2, D_3).$$

Decrypt. Any group member, say John, can decrypt Danny's encrypted photos as long as his secret key SK_J satisfies the embedded access tree T_D. As the underlying access tree T_D is constructed hierarchically. SK_J should first satisfy the basic access tree of the group T_{AG} which has been embedded into T_D as shown in Fig. 2. A recursive algorithm $DecryptBasicTree(x)$ is introduced as follows to check this.

1. For each leaf node $x \in T_D$, computes:

$$H_x = DecryptBasicTree(x) = \begin{cases} e(C_x, g^{\frac{I_A L_J P_x(0)}{t_x}}), att(x) \in \mathcal{B}_J \\ \perp, \text{ otherwise.} \end{cases}$$

2. For non-leaf node x, H_x will be computed recursively. For each of his child y, call $H_y = DecryptBasicTree(y)$. Let S_x be an arbitrary k_x-sized set of child nodes such that $H_y \neq \perp$. If no such set exists, $H_x = \perp$. Otherwise:

$$\begin{aligned} H_x &= \prod_{y \in S_x} H_y^{\Delta_{i,S'_x}(0)}, \begin{cases} i = index(y) \\ S'_x = \{index(y), y \in S_x\} \end{cases} \\ &= \prod_{y \in S_x} e(g^{t_y q}, g^{\frac{I_A L_J P_y(0)}{t_y}})^{\Delta_{i,S'_x}(0)} \\ &= \prod_{y \in S_x} e(g,g)^{q I_A L_J P_{parent(y)}(index(y)) \Delta_{i,S'_x}(0)} \\ &= \prod_{y \in S_x} e(g,g)^{q I_A L_J P_x(i) \Delta_{i,S'_x}(0)} \\ &= e(g,g)^{q I_A L_J P_x(0)}, \text{ (using polynomial interpolation).} \end{aligned}$$

Now, *DecryptBasicTree* has been completely defined and the algorithm calls *DecryptBasicTree* at the root node r:

$$DecryptBasicTree(r) = e(g,g)^{qI_AL_JP_x(0)} = e(g,g)^{vqI_AL_J}.$$

Next, the algorithm checks whether SK_J satisfies the entire T_D by calling another recursive algorithm $DecryptTree(x)$ defined as follows.

1. For each leaf node $x \in T_D$, computes:

$$H_x = DecryptTree(x) = \begin{cases} e(C'_x, g^{\frac{vI_AL_J}{s_j}}), att(x) \in \mathcal{A}_J \cup (\mathcal{B}_J \cap \mathcal{B}_G) \\ \perp, \text{ otherwise.} \end{cases}$$

2. For non-leaf node x, H_x will be computed recursively. For each of his child y, call $H_y = DecryptTree(y)$. Let S_x be an arbitrary k_x-sized set of child nodes who satisfy that $H_y \neq \perp$. If no such set exists, $H_x = \perp$. Otherwise,
   (a) AND Gate. x is an AND gate:

$$H_x = \prod_{y \in S_x} H_y = \prod_{y \in S_x} e(g^{s_y n_y}, g^{\frac{vI_AL_J}{s_y}}) = e(g,g)^{vI_AL_Jn_x}.$$

   (b) OR Gate. x is an OR gate and as long as there exists y such that $H_y \neq \perp$:

$$H_x = H_y = e(g,g)^{vI_AL_Jn_y} = e(g,g)^{vI_AL_Jn_x}.$$

Now, *DecryptTree* has been completely defined and the algorithm calls *DecryptTree* at the root node r:

$$DecryptTree(r) = e(g,g)^{vnI_AL_J}.$$

If John's secret key SK_J satisfies both T_{AG} and T_D, he will get:

$$\begin{aligned} Temp &= \frac{e(g^{\frac{\alpha + I_AL_J}{\beta}}, g^{\beta v(n+q)})}{DecryptBasicTree(r) \cdot DecryptTree(r)} \\ &= \frac{e(g^{\frac{\alpha + I_AL_J}{\beta}}, g^{\beta v(n+q)})}{e(g,g)^{vqI_AL_J} \cdot e(g,g)^{vnI_AL_J}} \\ &= e(g,g)^{\alpha v(n+q)}. \end{aligned}$$

Finally, John can decrypt Danny's photo as:

$$\frac{D_0}{Temp} = \frac{Me(g,g)^{\alpha v(n+q)}}{e(g,g)^{\alpha v(n+q)}} = M.$$

5 Security

5.1 Threat Model

The OSN-SP is a semi-trusted for-profit server which will honestly follow the designated protocol but might collect users' sensitive information for improper aims. The group leader is assumed to be trusted and will honestly distribute secret keys to his friends in the group, which is consistent with previous researches [8]. This is reasonable due to that we have to finally trust someone and the group leader is normally the friend of his group members. Members will feel more secure to trust their friends than the OSN-SP. Users of different groups cannot collude with each other due to the different basic access policies. Users within the same social group might collude with others and try to decrypt ciphertexts that are not meant for them.

5.2 Security Proof

To be consistent with related work [4] and [5], we adopt a formal security game model [11] between a challenger and an adversary Ad, called selective-set model. Here, the challenger plays the role of group leader. The adversary can be the OSN-SP or a malicious user. For a scheme to be semantically secure, any adversary Ad must not learn any information about the sensitive data. As the OSN-SP will honestly follow the designated protocol, we assume that the OSN-SP has successfully run **Prepare** and **Register** for the application and the game will be simulated from the third phase **ApplyToBuildGroup** as follows.

1. **ApplyToBuildGroup.** The challenger runs **ApplyToBuildGroup**.
2. **BuildGroup.** The challenger runs **BuildGroup** to build a group.
3. **Phase 1.** Ad performs repeated queries for the secret key SK_{Ad} in the group.
4. **Challenge.** Ad submits two equal length messages M_0 and M_1. In addition, Ad gives a challenge access tree T_{Ad} such that none of the secret keys generated in Phase 1 satisfies T_{Ad}. The challenger flips a random coin b and encrypts M_b under T_{Ad}. The ciphertext C is given to Ad.
5. **Phase 2.** Phase 1 is repeated with the restriction that none of the secret keys satisfies the access tree corresponding to the challenge.
6. **Guess.** Ad outputs a guess b' of b.

Definition 1. Masque is secure if all polynomial time adversaries have at most a negligible advantage in the above game.

Theorem 1. Let q be the total number of group elements the adversary Ad receives from the interactions in the above game. The advantage of Ad in the game is $\mathcal{O}(\frac{q^2}{p})$.

Proof. We will use the generic bilinear group model of [11]and [12] to argue that no efficient adversary that acts generically on the groups can break the security of Masque with any reasonable probability. Let f_1 and f_2 be random encoding functions:

1. f_1: A random encoding for group G_1. $g^{s_j} \in G_1$ will be denoted as $f_1(s_j)$.
2. f_2: A random encoding for group G_2. $e(g,g)^{\alpha v}$ will be denoted as $f_2(\alpha v)$.

In generic group model, group elements are encoded as unique random strings. An adversary cannot test any property other than equality. In order to simulate two different ciphertexts, we set the component D_4 in the challenge phase as either $f_2(\alpha v(n+q))$ or $f_2(\theta)$, where θ is a random value. *Ad* has to decide whether $D_4 = f_2(\alpha v(n+q))$ or $D_4 = f_2(\theta)$. The security game is simulated as follows.

1. **ApplyToBuildGroup.** The challenger runs **ApplyToBuildGroup** and gets receipt for the group R_{AG}.
2. **BuildGroup.** The challenger runs **BuildGroup** and generates PK_{AG} and MK_{AG} for the group.
3. **Phase 1.** *Ad* performs repeated queries for the secret key SK_{Ad} in the group.
4. **Challenge.** *Ad* submits two equal-length messages M_0 and M_1 and a challenge access tree T_{Ad} such that none of the secret keys generated in Phase 1 satisfies the access tree. The challenger flips a random coin b and encrypts M_b under T_{Ad}. The ciphertext C is given to *Ad*.
5. **Phase 2.** Phase 1 is repeated with the restriction that none of the secret keys satisfies the access tree corresponding to the challenge.
6. **Guess.** *Ad* outputs a guess b' of b.

Each random encoding is associated with a distinct rational function $f = \frac{\xi}{\omega}$ over the indeterminate variables during the game. ξ and ω are polynomial functions and the variables are the randomly picked elements during the game. Next we will show that *Ad* can distinguish $D_4 = f_2(\theta)$ and $D_4 = f_2(\alpha v(n+q))$ with the probability $O(\frac{q^2}{p})$.

First, we consider *Ad*'s behavior when we set $D_4 = f_2(\theta)$. When *Ad* makes query to the oracles, *Ad*'s behavior can change when an unexpected collusion happen. An unexpected collusion may happen due to the random choice of the variables. When two queries separately corresponding to rational functions $f = \frac{\xi}{\omega}$, $f' = \frac{\xi'}{\omega'}$ and $f \neq f'$, but due to the random choices of these variables, we have that the values of $\frac{\xi}{\omega}$ and $\frac{\xi'}{\omega'}$ coincide. If such collusion happens, the simulator quits and the adversary wins. Based on the Schwartz[13] lemma, the probability of the collusion is $O(\frac{1}{p})$. The advantage of *Ad* is $O(\frac{q^2}{p})$. The probability of no such collusion is $1 - O(\frac{q^2}{p})$. This probability is negligible and we assume that no such collusion happens.

Now we consider *Ad*'s behavior will be identically distributed even if we set $D_4 = f_2(\alpha v(n+q))$. Since we are in the generic group model where each group element's representation is uniformly and independently chosen, the only way that *Ad*'s behavior can be different is there are two queries q and q' into G_2 such that $q \neq q'$ but $q = q' = \alpha v(n+q)$. We will show this is impossible as *Ad* cannot construct a query into G_2 which coincides to be $e(g,g)^{\alpha v(n+q)}$. We will prove it by showing that none of *Ad*'s queries can be equal to $f_2(\alpha v(n+q))$.

We proceed by analyzing queries that Ad made into generic group oracles using the group elements received from the interactions. First we observe that Ad can make query which contains the term $f_2(\alpha v(n+q))$ by pairing $e(g^{\frac{\alpha+I_A L_D}{\beta}}, g^{\beta v(n+q)})$ to get $f_2(\alpha v(n+q) + I_A L_D v(n+q))$. To get only $f_2(\alpha v(n+q))$, Ad has to combine group elements received from the interactions to cancel $f_2(I_A L_D v(n+q))$. Ad must have all necessary secret key components to go through access tree $\mathcal{T}_{Ad}$ to obtain $f_2(vnI_A L_D)$ and $f_2(vqI_A L_D)$. However, this is impossible because in Phase1 and Phase2, Ad queries with the restriction that none of the secret keys in Phase 1 satisfies $\mathcal{T}_{Ad}$. Therefore, any adversary query polynomial of this form cannot be the form of $f_2(\alpha v(n+q))$ and all adversaries have at most a negligible advantage in the above game. According to **Definition 1**, Masque is secure.

5.3 Collusion-Resistance

Masque randomizes the group secret key SK of each member with a randomly chosen value L, which makes the combination of components in different users' SK meaningless. Namely, users cannot go through the embedded access tree by combining their secret key components to decrypt data that are out of their access scope.

6 Conclusion

In this paper, we have proposed a hierarchical fine-grained access control mechanism for interactive sharing of encrypted data in OSNs, called Masque, where the OSN-SP can specify basic access policies for social groups while members within a group could customize their own access policies and share data with potential friends without compromising security and privacy. Thorough analysis has shown that Masque is proven secure. As for the future work, we will consider how to achieve instantaneous user revocation so that users could flexibly re-customize specific access policies for their sensitive data.

References

1. Boneh, D., Franklin, M.K.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
2. Boneh, D., Boyen, X.: Efficient Selective-ID Secure Identity-Based Encryption without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 223–238. Springer, Heidelberg (2004)
3. Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
4. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. In: 13th ACM Conference on Computer and Communications Security (CCS 2006), pp. 89–98 (October 2006)

5. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-Policy Attribute-Based Encryption. In: 28th IEEE Symposium on Security and Privacy (S&P 2007), pp. 321–334 (May 2007)
6. Pirretti, M., Traynor, P., McDaniel, P., Waters, B.: Secure Attribute-Based Systems. In: 16th ACM Conference on Computer and Communications Security (CCS 2009), pp. 799–873 (November 2009)
7. Traynor, P., Kevin, B., Enck, W., McDaniel, P.: Realizing Massive-Scale Conditional Access Systems Through Attribute-Based Cryptosystems. In: 13th Network and Distributed System Security Symposium (NDSS 2008), pp. 799–873 (February 2008)
8. Badenand, R., Benderand, A., Spring, N., Bhattacharjee, B., Starin, D.: Persona: An Online Social Network with User-Defined Privacy. In: ACM Conference on Data Communication (SIGCOMM 2009), pp. 135–146 (August 2009)
9. Bobba, R., Fatemieh, O., Khan, F., Khanand, A., Gunter, C.A., Khurana, H., Prabhakaran, M.: Attribute-Based Messaging: Access Control and Confidentiality. In: ACM Conference on Data Communication (SIGCOMM 2009), vol. 13 (December 2010)
10. Yu, S., Wang, C., Ren, K., Lou, W.: Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing. In: 29th Conference on Computer Communications (INFOCOM 2010), pp. 14–19 (March 2010)
11. Shoup, V.: Lower Bounds for Discrete Logarithms and Related Problems. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 256–266. Springer, Heidelberg (1997)
12. Harney, H., Colgrove, A., McDaniel, P.: Principles of Policy in Secure Groups. In: 18th Network and Distributed System Security Symposium (NDSS 2011), pp. 66–74 (August 2011)
13. Schwartz, J.T.: Fast Probabilistic Algorithms for Verification of Polynomial Identities. Journal of the ACM 27, 701–717 (1980)

Mitigating the Intractability of the User Authorization Query Problem in Role-Based Access Control (RBAC)

Nima Mousavi and Mahesh V. Tripunitara

ECE, University of Waterloo, Canada
{nmousavi,tripunit}@uwaterloo.ca

Abstract. We address the User Authorization Query problem (UAQ) in Role-Based Access Control (RBAC) which relates to sessions that a user creates to exercise permissions. Prior work has shown that UAQ is intractable (**NP**-hard). We give a precise formulation of UAQ as a joint optimization problem, and observe that in general, UAQ remains in **NP**. We then investigate two techniques to mitigate its intractability. (1) We efficiently reduce UAQ to boolean satisfiability in conjunctive normal form, a well-known **NP**-complete problem for which solvers exist that are efficient for large classes of instances. We point out that a prior attempt is not a reduction, is inefficient, and provides only limited support for joint optimization. (2) We show that UAQ is fixed-parameter polynomial in the upper-bound set of permissions under reasonable assumptions. We discuss an open-source implementation of (1) and (2), based on which we have conducted an empirical assessment.

1 Introduction

We address the User Authorization Query problem (UAQ) in Role-Based Access Control (RBAC) [8,16,17]. UAQ arises in the context of RBAC sessions [9,14], and is known to be intractable [4,8]. We address how this intractability can be mitigated.

In RBAC, permissions are not assigned directly to users. Rather, a user is assigned to roles, which in turn are assigned permissions. A user is authorized to those permissions that are assigned to the roles to which he is authorized. In Figure 1, for example, a user is assigned to three roles, and is authorized to five permissions via those roles.

If a user wants to exercise a permission, he must first create a session [9,14]. He specifies the roles he would like associated with a session when creating it. A set of Separation of Duty (SoD) constraints[1], D, may be imposed on him. Each such constraint is of the form $\langle R, t\rangle$, where R is a set of roles and t is an integer such that $2 \leq t \leq |R|$. The semantics of such a constraint is that t or more roles from the set R are not allowed to be activated in any session. In the example in Figure 1, the constraint $\langle\{\text{Finance}, \text{Human Resources}, \text{Purchasing}\}, 3\rangle$ precludes the user from activating all three roles in the same session.

UAQ is an optimization problem with two distinct objectives. The user has a set of permissions to which he wants the session to be authorized. However, a set of roles

[1] Some prior work [12] calls these "mutually exclusive role constraints" to clearly distinguish the security objective (separation of duty) from the enforcement mechanism. We call them SoD constraints to be consistent with prior work on UAQ [8,16,17].

L. Xu, E. Bertino, and Y. Mu (Eds.): NSS 2012, LNCS 7645, pp. 516–529, 2012.

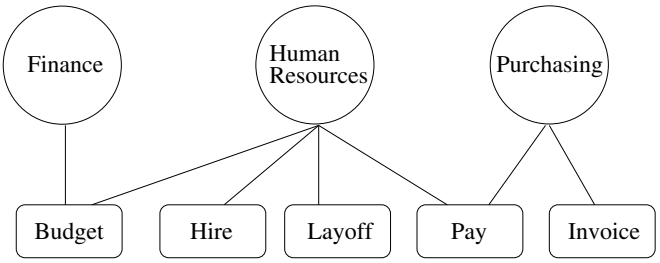


Fig. 1. An example of an RBAC policy for a user. Circles are roles and rectangles are permissions.

that give him exactly the permissions he seeks may not exist [17]. Consequently, the permissions that a user wants are specified as two sets [16]: a lower bound set P_{lb} and an upper bound set $P_{ub} \supseteq P_{lb}$. The permissions in P_{lb} are those to which the session must be authorized, and $P_{ub} - P_{lb}$ is a "slack" or extra set of permissions to which the session may be authorized. The user may wish to either minimize or maximize the number of extra permissions to which his session is authorized. His choice depends on which of two security objectives, safety or availability, he wants to prioritize over the other. If he prioritizes safety, then he would want to minimize the number of permissions from $P_{ub} - P_{lb}$. If he prioritizes availability, he would want to maximize the number of permissions from $P_{ub} - P_{lb}$. In the latter case, he still has safety in that the permissions cannot exceed P_{ub}.

Apart from the number of extra permissions that is an optimization objective as we discuss above, the number of roles that are activated is another optimization objective. We may want, for example, to minimize the number of roles that are activated in the session. This may be an important consideration for the system; minimal sets of roles in sessions may be more efficient for the system to support. It has been argued [17] that we may instead want to maximize the number of roles to which a session is authorized. As prior work [4] points out, from the standpoint of computational complexity, allowing both options of minimization and maximization is no more difficult than allowing minimization only. (We point out, however, that the assertion in [4] that constraints do not impact the computational complexity of UAQ is not true. With the introduction of constraints, if $\mathbf{P} \neq \mathbf{NP}$, the sub-case of UAQ that maximizes the number of extra permissions is no longer tractable.)

UAQ Specification. An instance of UAQ is specified by the following inputs.

- An RBAC policy, ρ, for a user, where $\rho = \langle RH, RP \rangle$, where RH is a role-hierarchy that relates roles in a partial order, and RP is an assignment of roles to permissions. For convenience, we assume that RH is reflexive, i.e., given a role r that appears in ρ, $\langle r, r \rangle \in RH$. We denote the set of all roles in ρ as $R[\rho]$, and the set of all permissions as $P[\rho]$.
- A set of SoD constraints, D, each of the form $\langle R, t \rangle$, where $t \in [2, |R|]$, $R \subseteq R[\rho]$.
- Two sets of permissions, P_{lb} and P_{ub}, with $P[\rho] \supseteq P_{ub} \supseteq P_{lb}$.

- An optimization objective for roles, $o_r \in \{\min, \max, \text{none}\}$. The option "none" means that we do not care to optimize the number of roles in a solution.
- An optimization objective for extra permissions, $o_p \in \{\min, \max, \text{none}\}$. If $P_{ub} = P_{lb}$, we assume $o_p = \text{none}$.
- A priority, $pri \in \{r, p\}$; pri indicates which of the two optimization objectives, roles or extra permissions, we prioritize over the other. The need for this arises from our two distinct optimization objectives. As we point out below in Example 2, without this parameter, there can exist two optimal solutions that are incomparable to one another. This parameter is meaningful only if $o_r \neq \text{none}$ and $o_p \neq \text{none}$. If $o_r = \text{none}$ and $o_p \neq \text{none}$, then $pri = p$, and if $o_p = \text{none}, o_r \neq \text{none}$, then $pri = r$. If both are none, then this input is ignored.

Prior work [17] also considers what are called cardinality constraints as an input. However, as subsequent work [16] points out, these can be addressed prior to formulating a UAQ instance. An instance may be associated with no valid solutions, or one or more valid solutions. A valid solution is $\langle R_s \rangle$, where $R_s \subseteq R[\rho]$ is a set of roles that satisfies every constraint in D, and the set of permissions $P_s \subseteq P[\rho]$, to which the roles in R_s are authorized, is such that $P_{ub} \supseteq P_s \supseteq P_{lb}$. If $o_r = \min$ (max), then $|R_s|$ is the minimum (maximum) possible number of roles. If $o_p = \min$ (max), then $|P_s - P_{lb}|$ is the minimum (maximum) possible number of extra permissions. More than one valid solution can exist. The decision version of UAQ that corresponds to the above optimization version:

- Does not take the inputs o_r, o_p and pri. Instead, it takes the following two inputs.
- $k_r \in \{=, \leq, \geq\} \times [1, |R|]$, where R is the set of roles in the policy ρ. This indicates the number of roles we seek in a solution, and whether that number is an exact, upper or lower bound.
- $k_p \in \{=, \leq, \geq\} \times [0, |P_{ub} - P_{lb}|]$. k_p is the extra-permissions analogue to k_r.

A decision instance is either true or false. It is true if there exists a set of roles that satisfies k_r and k_p, and false otherwise.

From the Decision to the Optimization Version. The decision and optimization versions of UAQ are related closely. There exists a polynomial-time Turing reduction [5] from the optimization version to the decision version. That is, given an oracle Ω for the decision version, we can efficiently solve the optimization version. A concrete approach is two-dimensional binary search. For example, for the case that $pri = p$ (i.e., prioritize permissions over roles), we first fix k_r at $\langle \geq, 1 \rangle$; i.e., we accept a solution with any number of roles. We then perform a binary search for the optimal number of permissions with $O(\log |P_{ub} - P_{lb}|)$ invocations to Ω. Once we find the optimal number of permissions, π, we then perform a binary search with $O(\log |R[\rho]|)$ invocations to Ω with k_p set to $\langle =, \pi \rangle$. This is exactly the approach we have implemented [13].

Also, given an oracle for the decision version that outputs 'true' or 'false,' a polynomial-time Turing reduction [5] can give us a certificate (i.e., set of roles) that is a solution. Indeed, a certificate is a by-product of tools such as SAT solvers along with the 'satisfiable' or 'unsatisfiable' output. We can then repeatedly look for additional solution sets of roles by excluding a role from $R[\rho]$ that is in a prior solution.

In summary, the optimization problem and the problem of identifying one or multiple sets of roles that are valid solutions all rely on the construction of Ω, an approach to the decision version. Consequently, we focus on the decision version of UAQ in the remainder of this paper.

Computational Complexity. UAQ is known to be intractable [4,8]. The sub-case with $P_{lb} = P_{ub}$, $D = \emptyset$ and $o_r =$ min has been shown to be **NP**-hard by Du and Joshi [8]. The sub-case with $D = \emptyset$, $o_r =$ none and $o_p =$ min has been shown to be **NP**-hard by Chen and Crampton [4]. Given those results for sub-cases, the general case is also **NP**-hard. However, the following theorem establishes an upper-bound on the general case; the decision version remains in **NP**. (Note that the "in **NP**" results from prior work [4,8] are for the sub-cases only.)

Theorem 1. *The decision version of UAQ* $\in$ ***NP****.*

Proof. It suffices to show that for every instance of UAQ, there is a polynomial-sized certificate that can be verified in polynomial time. Let R be the set of all roles in the RBAC policy, ρ. A certificate is a set of roles R', where $R' \subseteq R$. This certificate is clearly polynomial (linear) in the size of the instance. Let P' be the set of permissions to which R' is authorized. An algorithm to verify the certificate first generates P' and then checks that $R' \subseteq R$, $|R'|$ satisfies k_r, $P_{lb} \subseteq P' \subseteq P_{ub}$, $|P' - P_{lb}|$ satisfies k_p, and R' satisfies every constraint in D. This algorithm is polynomial-time in the instance.

Our Work. We consider how the intractability of UAQ can be mitigated. That is, how we can efficiently address instances of UAQ that may arise, notwithstanding its worst-case intractability. We seek approaches that have three properties: soundness, efficiency and full support for joint-optimization. As we discuss in Section 2, prior approaches do not satisfy one or more of these properties.

Soundness means that a solution output by an approach to the optimization problem must be valid. A solution output by an approach to the decision problem must be correct. Efficiency refers to the above theorem that establishes that **NP** is an upper-bound for the inefficiency of any approach. In particular, it is known that **NP** $\subseteq$ **PSPACE** $\subseteq$ **EXP**[2], and both containments are thought to be strict [3]. An approach that causes an exponential blowup in its design can be considered to be efficient only if **NP** $=$ **PSPACE** $=$ **EXP**. Full support for joint optimization means that an approach should allow us to minimize or maximize the number of roles independently of minimizing or maximizing the number of extra permissions. That is, the two optimization objectives are distinct, and an approach must recognize this.

We investigate two standard, theoretically well-founded approaches: efficient reduction to CNF-SAT[3] and fixed-parameter tractability. For each, we provide algorithms and an empirical assessment.

[2] **PSPACE** is the class of decision problems that can be solved given space polynomial in the size of the input. **EXP** is the class of decision problems that can be solved given time exponential in the size of the input.

[3] CNF-SAT is the problem of deciding whether a boolean expression in Conjunctive Normal Form (CNF) is satisfiable. It is a well-known **NP**-complete problem [10].

Example 1. Consider the RBAC policy for a user, ρ, from Figure 1. Let $D = \{\langle\{\text{Human Resources}, \text{Purchasing}\}, 2\rangle\}$; that is, the only SoD constraint is that the roles Human Resources and Purchasing are not allowed to be activated in the same session. Let $P_{lb} = \{\text{Pay}\}$. We consider several example requests with different values for the other parameters.

- $o_p = \text{min}, P_{ub} = P_{lb}$ and any values for the other inputs: has no valid solutions because any choice for R_s results in more permissions than Pay only.
- $o_p = \text{min}, P_{ub} = P_{lb} \cup \{\text{Hire}, \text{Invoice}\}$ and any values for the other inputs: the only valid solution is $R_s = \{\text{Purchasing}\}$, with one extra permission, Invoice.
- $o_p = \text{max}, o_r = \text{min}, pri = p, P_{ub} = P_{lb} \cup \{\text{Budget}, \text{Hire}, \text{Layoff}, \text{Invoice}\}$: the only valid solution is $R_s = \{\text{Human Resources}\}$.
- If we change pri to r in the previous example request, the valid solution remains $R_s = \{\text{Human Resources}\}$. We choose Human Resources over Purchasing, even though both result in only 1 role, because of the secondary requirement of maximizing extra permissions.

Example 2. This example illustrates the need for the input pri. Suppose ρ is the policy from Figure 1, $P_{lb} = \{\text{Budget}, \text{Pay}\}$, P_{ub} is all the permissions from the figure, $D = \emptyset$, $o_p = \text{min}$ and $o_r = \text{min}$.

- If $pri = r$, then our solution is $\{\text{Human Resources}\}$. We have only 1 role and 2 extra permissions (Hire and Layoff).
- If $pri = p$, then our solution is $\{\text{Finance}, \text{Purchasing}\}$. We have 2 roles and only 1 extra permission (Invoice).

If we do not have the input pri, the solutions would both be valid, but incomparable to one another.

Layout. In the next section, we discuss related work. In that context, we discuss also soundness, efficiency and joint-optimization issues with prior approaches. In Section 3, we reduce UAQ to CNF-SAT. In Section 4, we discuss an algorithm for UAQ that is fixed-parameter polynomial-time. We present empirical results in Section 5 and conclude with Section 6.

2 Related Work

To our knowledge, UAQ was first proposed by Du and Joshi [8]. That work shows also that the subcase of optimizing roles is **NP**-hard and the corresponding decision version is in **NP**. That work does not consider constraints or the optimization of extra permissions. Zhang and Joshi [17] generalize UAQ by introducing the problem of optimizing the number of extra permissions in addition to the number of roles. That work also introduces constraints. Wickramaarachchi et al. [16] propose two approaches for mitigating the intractability of UAQ. That work also proposes a different combination of mixing the optimization of number of roles and permissions than Zhang and Joshi [17]. Chen and Crampton [4] consider the subcase of UAQ that is the optimization of permissions as a problem that is related to a variant of the well-known set cover problem [10].

That work establishes that the minimization version of the subcase it considers is **NP**-hard and the maximization version is in **P**. That work does not consider constraints or the optimization of roles.

To our knowledge, the pieces of prior work that consider the mitigation of the intractability of UAQ are those of Du and Joshi [8], Zhang and Joshi [17], Wickramaarachchi et al. [16] and Armando et al. [2]. The issues with the approaches approaches of Du and Joshi [8] and Zhang and Joshi [17] are identified by Wickramaarachchi et al. [16] and we refer the reader to that work for a comprehensive discussion. We provide a summary here. The work of Du and Joshi [8] considers only the problem of role minimization in the absence of constraints. The approach is inefficient as it is exponential-time in its design. The work of Zhang and Joshi [17] first proposes a greedy algorithm that is not sound [16]. Their approach to dealing with the unsoundness renders the algorithm inefficient; it is exponential-time in its design.

The work of Armando et al. [2] points out, based on empirical observations, that the approach of Wickramaarachchi et al. [16] is inefficient. They then propose a complementary approach of caching session information for greater efficiency. That work does not identify the inefficiency inherent to the approach of Wickramaarachchi et al. [16] as we do below. Their work is complementary to our work. The work of Wickramaarachchi et al. [16] proposes two approaches to mitigate the intractability of UAQ. Both leverage prior work on boolean satisfiability (SAT). The first approach is an algorithm similar to an earlier algorithm for deciding SAT instances. The algorithm is exponential-time in its design. This analytical observation is validated by their empirical observations, which show that this first approach of theirs is inefficient in comparison to the second. The approach also does not address the joint optimization.

The second approach proposes a mapping to CNF-SAT (SAT instances in Conjunctive Normal Form). The problem of deciding whether a boolean expression in CNF is satisfiable is known to be **NP**-complete [10]. Unfortunately, this second approach is unsound, inefficient and is limited in the manner in which the joint optimization is addressed. The unsoundness arises from the manner in which the RBAC policy ρ is mapped. Suppose $r \in R[\rho]$ is a role such that $\{\langle r, r_1\rangle, \ldots, \langle r, r_k\rangle\} \subseteq RH$, and $\{\langle r, p_1\rangle, \ldots, \langle r, p_m\rangle\} \subseteq RP$. Then the approach proposes that we encode this as a clause $r \leftrightarrow r_1 \wedge \ldots \wedge r_k \wedge p_1 \wedge \ldots \wedge p_m$, where "$\leftrightarrow$" is "if and only if" and "$\wedge$" is conjunction. (We abuse notation slightly in our use of r_i, p_j as both roles and permissions, and boolean variables that correspond to them.)

The problem with this is that there can exist UAQ instances for which the approach incorrectly determines that there is no valid solution. Given a constraint $\langle R, t\rangle$, if a permission $p \in P_{lb}$ is assigned to t or more roles in R, then the approach would deem that there is no valid solution, which is incorrect. This problem is quite general, and not some small corner-case. For every constraint $\langle R, t\rangle$, there exists an infinite number of UAQ instances for which the approach is unsound.

One may be tempted to adopt a "quick fix" to address the above problem. For example, one may change the "if and only if" to an "only if" only. This will not work. The reason is that the manner in which the joint optimization is addressed by Wickramaarachchi et al. [16] relies on the "if and only if." We discuss this further below in the context of the limited support for joint optimization in that work.

The approach of Wickramaarachchi et al. [16] is inefficient. The inefficiency arises from the manner in which constraints are encoded. Given a constraint $\langle R, t \rangle$, the approach first enumerates every t-sized subset of R. Suppose $\{r_1, \ldots, r_t\}$ is such a subset. The approach encodes the subset as a clause $\neg r_1 \vee \ldots \neg r_t$. The problem with this is that it results is an exponential blowup in its design. As with unsoundness, there is an infinite number of classes of UAQ instances, with infinite members in each, for which this approach causes an exponential blowup. An example is when t is polynomial in $|R|$, e.g., $t = \sqrt{|R|}$.

Finally, the approach offers only limited support for the joint optimization of the numbers of roles and extra permissions. This issue is related to the unsoundness above. The manner in which joint optimization is supported is using the notion of relaxable and non-relaxable clauses. A clause is said to be non-relaxable if in a truth-assignment, that clause must evaluate to 1; otherwise it is relaxable. There exist SAT solvers that, in addition to meeting the constraint on non-relaxable clauses, maximize the number of relaxable clauses that evaluate to 1 in a truth-assignment.

The idea in Wickramaarachchi et al. [16] to minimize the number of roles, then, is to add a clause $\neg p$ for each $p \in P_{ub} - P_{lb}$ and specify that every such clause is relaxable. To maximize the number of roles, we add a clause p (rather than $\neg p$) and specify those to be relaxable. In conjunction with the "if and only if" clauses we discuss above in the context of unsoundness, this ensures that the number of roles is maximized if the number of extra permissions is maximized, and the number of roles is minimized if the number of extra permissions is minimized. Unfortunately, there is no obvious way to address other combinations, for example, minimize the number of roles and maximize the number of extra permissions.

3 Efficient Reduction to CNF-SAT

In this section, we discuss the first of our oracles for the decision version of UAQ — a polynomial-time many-one reduction [3] to CNF-SAT. The main technical challenges are to capture the SoD constraints and multiple objectives (roles and extra permissions).

An instance of CNF-SAT is a set of clauses that are a conjunction. A clause comprises one or more literals that are a disjunction. A literal is a variable or its negation. For each role $r_i \in R$ and permission $p_j \in P_{ub}$, we define a boolean variable which is true if and only if r_i or p_j is activated, respectively. A satisfying assignment corresponds to a valid solution. If a SAT solver discovers that an input instance is satisfiable, as auxiliary output, it provides an assignment to the variables, which immediately tells us the set of roles to be activated. (Even if it does not, it is easy for us to construct one using a polynomial-time Turing reduction [3].)

The RBAC Policy, ρ, and Permission Sets, P_{lb} and P_{ub}. We adopt the approach of prior work [16], with a correction for their soundness issue, to capture ρ and P_{lb} in CNF-SAT. That is, for each $\langle r, p \rangle \in RP$ we capture it as a clause $r \rightarrow p$, where "$\rightarrow$" is "implies." Also, for every permission p, if $r_1, \ldots, r_n$ are the roles to which it is authorized in ρ, we add a clause $p \rightarrow r_1 \vee \ldots \vee r_n$. These capture our intent that if r is activated, p is activated, and for p to be activated, at least one of the roles to which it is authorized

must be activated. We also capture each $\langle r_1, r_2\rangle \in RH$ with a clause $r_1 \rightarrow r_2$. This encodes the requirement that a junior role must be activated if a senior role is. This is optional — Wickramaarachchi et al. [16] provide a discussion of how we can deal with RH in the context of UAQ. We refer the reader to that work for a discussion.

We add a clause p for each $p \in P_{lb}$, i.e., a clause with the single variable p. This captures our intent that every permission in P_{lb} must be activated. For every permission in $P[\rho] - P_{ub}$, we add the clause $\neg p$. This captures our intent that only permissions in P_{ub} are allowed to be activated. (We can also simply remove those permissions and any role that is authorized to any of those permissions before formulating our UAQ instance.)

SoD Constraints. This is one of the technical challenges with reducing UAQ to SAT. We adopt an approach similar to that of Sinz [15]. We first encode each constraint as a boolean circuit. That is, we first reduce constraint-satisfaction to CIRCUIT-SAT (the satisfiability problem for boolean circuits), and then adopt a "textbook" reduction from CIRCUIT-SAT to CNF-SAT [6].

We first describe boolean circuits that we call Bit-Sum, Sum and Compare that are building blocks. Then, we propose a circuit that we call Max-Circuit. It takes input n bits and an integer k, and outputs 1 if and only if at most k of the n bits are 1. All inputs to a boolean circuit are bits; an integer input is encoded in binary. Our encoding of a constraint $\langle R_c, t\rangle \in D$ is as a Max-Circuit with input $\langle r_1, \ldots, r_n\rangle$ and t, where $R_c = \{r_1, \ldots, r_n\}$.

- **Bit-Sum**
  - Input: $\langle y_n, y_{n-1}, \ldots, y_1\rangle$, x
  - Output: $\langle z_n, z_{n-1}, \ldots, z_1\rangle$, such that $\sum_{i=1}^{n} z_i 2^{i-1} = \sum_{i=1}^{n} y_i 2^{i-1} + x$.
  - Let $c_1, \ldots, c_{n-1}$ be carry variables. Then, $z_1 = x \oplus y_1$, $c_1 = x \wedge y_1$ and $z_i = c_{i-1} \oplus y_i$, $c_i = c_{i-1} \wedge y_i$ for all $i \in \{2, \ldots, n\}$. The number of gates in a Bit-Sum circuit is $2n$.
- **Sum**
  - Input: $\langle x_n, x_{n-1}, \ldots, x_1\rangle$
  - Output: $\langle z_m, z_{m-1}, \ldots, z_1\rangle$, such that $\sum_{i=1}^{m} z_i 2^{i-1} = \sum_{i=1}^{n} x_i$, $m = \lceil \log n \rceil$.
  - The Sum circuit uses n modules of Bit-Sum, each of m gates. The total number of gates in Sum is $2nm = 2n\lceil \log n \rceil$. Sum = Bit-Sum$(x_n$, Bit-Sum$(x_{n-1}, \ldots,$ Bit-Sum$(x_1, y_m, y_{m-1}, \ldots, y_1)\ldots))$, where $\langle y_m, y_{m-1}, \ldots, y_1\rangle = \langle 0, 0, \ldots, 0\rangle$.
- **Compare**
  - Input: $(x_n, x_{n-1}, \ldots, x_1)$, $(y_n, y_{n-1}, \ldots, y_1)$
  - Output: z, such that $z = 1$ if and only if $\sum_{i=1}^{n} x_i 2^{i-1} \leq \sum_{i=1}^{n} y_i 2^{i-1}$.
  - Let $e_1, \ldots, e_n$ and $l_1, \ldots, l_n$ be equality and less-ness variables, respectively. Then $e_n = x_n \leftrightarrow y_n$, $l_m = \neg x_n \wedge y_n$ and $e_i = e_{i+1} \wedge (x_i \leftrightarrow y_i)$, $l_i = ((\neg x_i \wedge y_i) \wedge e_{i+1}) \vee l_{i+1}$ for all $i \in \{1, \ldots, n-1\}$. Then $z = l_1 \vee e_1$. The number of gates in Compare is $5n - 2$.
- **Max-Circuit**
  - Input: $(x_n, x_{n-1}, \ldots, x_1)$, $k \leq n$
  - Output: z, such that z equals to 1 if and only if the number of true variables in $x_n, x_{n-1}, \ldots, x_1$ is at most k.

- Let $m = \lceil \log n \rceil$ and $y_m, \ldots, y_1$ be such that $\sum_{i=1}^{m} y_i 2^{i-1} = k$. Then $z = \text{Compare}(\text{Sum}((x_n, x_{n-1}, \ldots, x_1)), (y_m, \ldots, y_1))$. The total number of gates in Max-Circuit(n) is at most $(2n + 5)\lceil \log n \rceil$.

Joint Optimization. This is the other technical challenge with UAQ. We use boolean circuits for this as well. We first introduce an additional circuit that we call Min-Circuit.

– **Min-Circuit**
  - Input: $(x_n, x_{n-1}, \ldots, x_1)$, $k \leq n$
  - Output: z, such that z equals to 1 if and only if the number of true variables in $x_n, x_{n-1}, \ldots, x_1$ is at least k.
  - $z = \text{Compare}((y_m, \ldots, y_1), \text{Sum}((x_n, x_{n-1}, \ldots, x_1)))$ and $\sum_{i=1}^{m} y_i 2^{i-1} = k$. The total number of gates in Min-Circuit(n) is at most $(2n + 5)\lceil \log n \rceil$.

To encode $k_r = \langle \leq, c \rangle$, i.e., that we want a set of roles of size at most c, we employ Max-Circuit with inputs $R[\rho]$ (the set of all roles in the RBAC policy) and c. To encode $k_r = \langle \geq, c \rangle$, we employ Min-Circuit with inputs $R[\rho]$ and c. To encode $k_r = \langle =, c \rangle$, we employ both. That is, each of Max- and Min-Circuit is eventually encoded as clauses in CNF-SAT, which are then conjuncted, thus giving us the semantics we seek with "=." We can similarly encode k_p, the decision version of the optimization objective for extra permissions.

Cost of the Reduction. Each gate of Max-Circuit and Min-Circuit is converted to 4 clauses. For $R = R[\rho]$ and $d(p) = |\{r_i : \langle r_i, p \rangle \in PA\}|$, the total number of clauses is $|P_{lb}| + \sum_{p \in P_{ub} - P_{lb}} (d(p) + 1) + \sum_{\langle R_c, t \rangle \in D} (8|R_c| + 20) \log |R_c| + (8|P_{ub} - P_{lb}| + 20) \log |P_{ub} - P_{lb}|$ which is $O(|P_{ub}|(|R| + \log |P_{ub}|) + |D||R| \log |R|)$.

That is, if the UAQ instance is of size n, our reduction outputs $O(n^2 \log n)$ clauses. We infer this from the term $|D|\,|R| \log |R|$, which dominates in the expression above — both D and R are input to the problem. As each clause can be of size $O(n)$, the running time of our reduction, and the size of the output CNF-SAT instance is $O(n^3 \log n)$.

4 Fixed-Parameter Tractability

The theory of fixed-parameter tractability [7] examines which parameters of a problem cause intractability. The question it considers is whether a problem becomes tractable if some parameter is bounded by a constant. This approach is particularly suited to UAQ as it is a problem with a number of input parameters. A problem is said to be *fixed-parameter polynomial* in a parameter k if there exists a polynomial-time algorithm for it given that k is bounded by some constant.

Before we make our assertion regarding fixed-parameter tractability, we discuss the algorithm in Figure 2. The algorithm imposes an upper bound of $|P_{ub}|$ on the number of roles in a solution set R_s. For each permission in P_{ub}, it picks a role as specified in Line 2–4. F contains all such sets of roles. As we adopt at most one role per permission, there are at most $(|R[\rho]| + 1)^{|P_{ub}|}$ sets in F. If we impose the constraint that $|R_s| \leq |P_{ub}|$, F contains every possible solution set of roles. In Lines 5–10, for each set of roles in F, we check if it satisfies the other parameters for a solution, namely k_r, k_p and D.

Let $R_p = \{r_j : r_j \text{ activates } p\}$ for any $p \in P[\rho]$;
Let F be the set of all sets $\{r_1\} \cup \cdots \cup \{r_{|P_{lb}|}\} \cup \{s_1\} \cup \cdots \cup \{s_{|P_{ub}-P_{lb}|}\}$ where:;
- $r_i \in R_p$ for some $p \in P_{lb}$, and,;
- $s_i \in R_{p'} \cup \{\epsilon\}$ for some $p' \in P_{ub} - P_{lb}$;

foreach $S \in F$ **do**
  Let S' be the set such that $r \in S'$ if and only if:;
    - either $r \in S$, or r is junior to some $r' \in S'$;
  $P_{actv} \leftarrow \{p : R_p \cap S' \neq \emptyset\}$;
  if P_{actv} *satisfies* k_p *and* S' *satisfies* k_r **then**
    if *checkD(*S'*)* **then return** S';
return $\emptyset$;

Fig. 2. An algorithm for the decision version of UAQ. It returns a set of roles that is a valid solution. The algorithm is exponential in $|P_{ub}|$, but is polynomial-time if $|P_{ub}|$ is bounded by a constant. We use "ϵ" in Line 4 to represent "no role." That is, in Line 4, s_i is either assigned to a role from $R_{p'}$, or to nothing. We assume access to an auxiliary routine, *checkD*, that checks whether its input set of roles satisfies every constraint in D. S' is needed only if we seek to activate all junior roles of a role that is activated. If not, we need S only, and the check in Line 10 is for S, not S'. If we seek all solutions, then rather than returning in Line 10, we continue to process all sets in F. F is guaranteed to contain all possible solutions.

Theorem 2. *Suppose the solution we seek, R_s, is such that $|R_s| \leq |P_{ub}|$. Then, UAQ is fixed-parameter polynomial in $|P_{ub}|$.*

Proof. The algorithm in Figure 2 is correct, and runs in time $O(n^{|P_{ub}|+2})$, where n is the size of the input other than P_{ub}. The reason is that $|F| \leq (|R[\rho]| + 1)^{|P_{ub}|} = O(n^{|P_{ub}|})$, and the processing of each set in F (e.g., the check against each constraint in D in Line 10) takes at worst quadratic time. Therefore, the algorithm is polynomial-time if $|P_{ub}| \leq c$ for some constant c.

We argue that the precondition in the assertion of the above theorem, that $|R_s|$ is bounded by $|P_{ub}|$, is reasonable, particularly when we seek to minimize $|R_s|$. For each permission, we expect to acquire it with one role. If $|R_s| > |P_{ub}|$, then we know that we have some redundant roles that can be removed without affecting the permissions to which the session is authorized. The theorem and corresponding algorithm are of interest because part of the motivation for UAQ is the principle of least privilege. There may be cases in which it is reasonable to assume that $|P_{ub}|$ is small. We include an oracle based on this algorithm in our empirical assessment in the next section.

5 Empirical Evaluation

We have implemented both our reduction to CNF-SAT (Section 3) and fixed parameter polynomial (Section 4) approaches. For the former, we produce input for the zChaff [1] SAT solver. For the optimization version, we have implemented the two-dimensional binary search that we discuss in Section 1. All our implementations are available for public download [13]. We have also made available our code for the generation of the

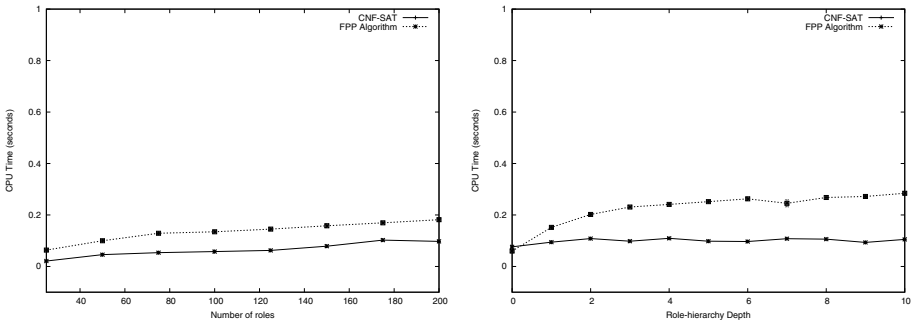


Fig. 3. Performance of the optimization versions of our approaches for different values of $|R[\rho]|$ and depth of the role hierarchy

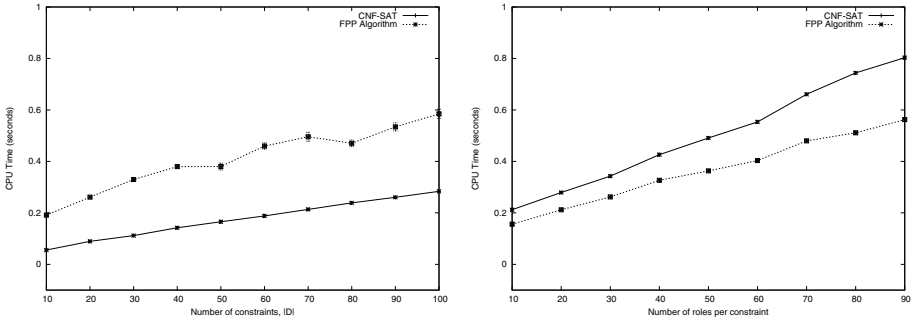


Fig. 4. Performance of the optimization versions of our approaches for different numbers of constraints and number of roles in constraints

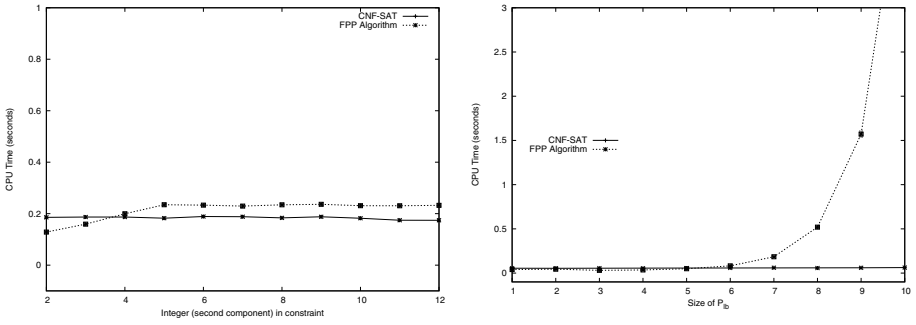


Fig. 5. Performance of the optimization versions of our approaches for different values of the integer (second component) of constraints and $|P_{lb}|$

test cases we have used in the empirical evaluation that we discuss in this section. We are not aware of any benchmark for UAQ. Consequently, we have done what prior work [8,16,17] does — generate several different kinds of test cases that exercise various parameters to the problem.

All the CPU times that we report are for full joint-optimization. As we perform binary search, the corresponding decision instances are about an order of magnitude faster. Our evaluations were conducted on a standard desktop PC with an Intel Dual Core E8400 CPU, each of which clocks at 3 GHz and has a 6 MB cache. The machine runs the 32-bit Ubuntu Linux 10.04 LTS operating system and has a 4GB RAM. All data points in our graphs represent a mean across at least 10 different inputs generated randomly, that correspond to the value in the horizontal axis, each of which was run 10 times. That is, each data point is an average across 100 runs. We also computed a 95% confidence interval which we show in the graphs with a vertical line segment at the data point. (These may be barely visible because the intervals are small.)

Overall Observations. Our approaches are orders of magnitude faster than prior approaches for which empirical results have been reported [16]. Furthermore, we are able to push our implementations far beyond prior approaches. For example, we have tried for up to $|R[\rho]| = 200$; prior work [16] goes only up to fewer than 75. Our results are not surprising to us. They are a consequence of the efficiency inherent to our reduction (in the case of the CNF-SAT approach), and the fact that the fixed-parameter polynomial-time algorithm indeed demonstrates efficient (polynomial-time) behaviour so long as $|P_{ub}|$ is bounded by a somewhat small constant.

We were given access to the decision version of the CNF-SAT approach of [16]. We tried several inputs for comparison. We discovered that its exponential behaviour becomes quickly apparent. For an RBAC policy of only 14 roles and a single SoD constraint in D, the approach from that work takes 1.5 minutes. For the same policy, our approach takes 0.005 seconds.

Specific Observations. We report CPU times along 6 different axes for both our approaches.

- Figure 3 shows our performance for different number of roles (i.e., $|R[\rho]|$) and depth of role-hierarchy. We observe that both our approaches are resilient to an increase in either parameter. We have tried up to a somewhat unrealistic role-hierarchy depth of 10; prior benchmarks [11] suggest that the maximum role-hierarchy in enterprise settings is 5.
- Figure 4 shows our performance for different number of constraints (i.e., $|D|$) and roles in a constraint (i.e., the first component of a constraint). Both approaches show a slow, linear worsening of performance in both cases. However, even for up to $|D| = 200$, the CNF-SAT approach takes less than 0.2 seconds. We have also tried up to a somewhat unrealistic number of roles in a constraint of 90. The CPU time for both our approaches remains less than 1 second.
- Figure 5 shows our performance for different values of the integer (second component) of constraints, and $|P_{lb}|$. We point out that as $P_{lb} \subseteq P_{ub}$, and therefore $|P_{lb}|$ is a lower-bound for $|P_{ub}|$. Both our approaches remain highly resilient to

different values of the integer in constraints. For increasing $|P_{lb}|$, however, the fixed-parameter polynomial-time algorithm starts to demonstrate exponential behaviour once the parameter crosses a particular small threshold value. This is completely expected; the algorithm is exponential in $|P_{ub}|$, and therefore demonstrates polynomial-time behaviour only if $|P_{ub}|$ is bounded. In our implementation, this bound is approximately 7. The CNF-SAT approach, on the other hand, remains highly efficient for the larger values of $|P_{lb}|$.

6 Conclusion and Future Work

We have addressed the User Authorization Query (UAQ) problem in RBAC — a joint optimization problem of identifying optimal roles and extra permissions for an RBAC session. We have identified several issues with prior work on this problem related to soundness, efficiency and limited support for the joint optimization. We have adopted a systematic approach based on the observations that the decision version of the general case remains in **NP**, and an effective approach to the decision version gives us an effective approach to the optimization version. We have then investigated two standard, theoretically well-founded approaches to addressing the decision version.

One is reduction to CNF-SAT that permits us to leverage existing SAT solvers. The other is identification that the problem is fixed-parameter polynomial in the upper-bound set of permissions under reasonable assumptions. We have implemented (1) and (2), and provided an empirical assessment that validates our analytical insights. All our code and data is available as open-source [13].

We have also some results regarding the approximability of UAQ. We have a negative result that unless **P = NP**, there exists no efficient algorithm that can approximate the number of roles within a factor of $n^{1-\epsilon}$, for any $\epsilon > 0$, where n is the number of roles. Also, it is **NP**-hard to minimize the number of extra permissions with an approximation ratio better than $O(\log n)$, where n is the number of extra permissions. We do not include these results in this paper owing to lack of space.

As future work, it will be useful to take a closer examination of the sources of complexity of UAQ. These may suggest alternate parameters for which UAQ is fixed-parameter tractable. It is possible, in tandem with the evolution of a benchmark for UAQ, that such approaches are deemed to be efficient in practice. Also of interest is the identification and evolution of a benchmark for UAQ as the basis for empirical assessments.

Acknowledgements. We thank the authors of [16] for making their implementation available to us. We thank Ninghui Li for reviewing an earlier draft of this paper.

References

1. zChaff (April 2012), `http://www.princeton.edu/~chaff/zchaff.html`
2. Armando, A., Ranise, S., Turkmen, F., Crispo, B.: Efficient run-time solving of RBAC user authorization queries: Pushing the envelope. In: Proceedings of the ACM Conference on Data and Applications Security and Privacy (CODASPY 2012). ACM (February 2012)

3. Arora, S., Barak, B.: Computational Complexity: A Modern Approach. Cambridge University Press (2009)
4. Chen, L., Crampton, J.: Set Covering Problems in Role-Based Access Control. In: Backes, M., Ning, P. (eds.) ESORICS 2009. LNCS, vol. 5789, pp. 689–704. Springer, Heidelberg (2009)
5. Cook, S.A.: The complexity of theorem-proving procedures. In: Proceedings of the Third Annual ACM Symposium on Theory of Computing, STOC 1971, pp. 151–158 (1971)
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 3rd edn. The MIT Press (September 2009)
7. Downey, R.G., Fellows, M.R.: Fixed-parameter tractability and completeness I: Basic results. SIAM Journal on Computing 24(4), 873–921 (1995)
8. Du, S., Joshi, J.B.D.: Supporting authorization query and inter-domain role mapping in presence of hybrid role hierarchy. In: Proceedings of the ACM Symposium on Access Control Models and Technologies, SACMAT 2006, pp. 228–236. ACM, New York (2006)
9. Ferraiolo, D.F., Sandhu, R.S., Gavrila, S., Kuhn, D.R., Chandramouli, R.: Proposed NIST standard for role-based access control. ACM Transactions on Information and Systems Security 4(3), 224–274 (2001)
10. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1990)
11. Komlenovic, M., Tripunitara, M., Zitouni, T.: An empirical assessment of approaches to distributed enforcement in role-based access control (rbac). In: Proceedings of the First ACM Conference on Data and Application Security and Privacy, CODASPY 2011, pp. 121–132. ACM, New York (2011)
12. Li, N., Tripunitara, M.V., Bizri, Z.: On mutually exclusive roles and separation-of-duty. ACM Trans. Inf. Syst. Secur., 10 (May 2007)
13. Mousavi, N., Tripunitara, M.V.: CNF-SAT and Fixed-Parameter Polynomial-Time Implementations for UAQ (April 2012), `https://ece.uwaterloo.ca/~tripunit/uaq/`
14. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. IEEE Computer 29(2), 38–47 (1996)
15. Sinz, C.: Towards an Optimal CNF Encoding of Boolean Cardinality Constraints. In: van Beek, P. (ed.) CP 2005. LNCS, vol. 3709, pp. 827–831. Springer, Heidelberg (2005)
16. Wickramaarachchi, G.T., Qardaji, W.H., Li, N.: An efficient framework for user authorization queries in rbac systems. In: Proceedings of the ACM Symposium on Access Control Models and Technologies, SACMAT 2009, pp. 23–32. ACM, New York (2009)
17. Zhang, Y., Joshi, J.B.D.: Uaq: a framework for user authorization query processing in rbac extended with hybrid hierarchy and constraints. In: Proceedings of the ACM Symposium on Access Control Models and Technologies, SACMAT 2008, pp. 83–92. ACM, New York (2008)

Author Index