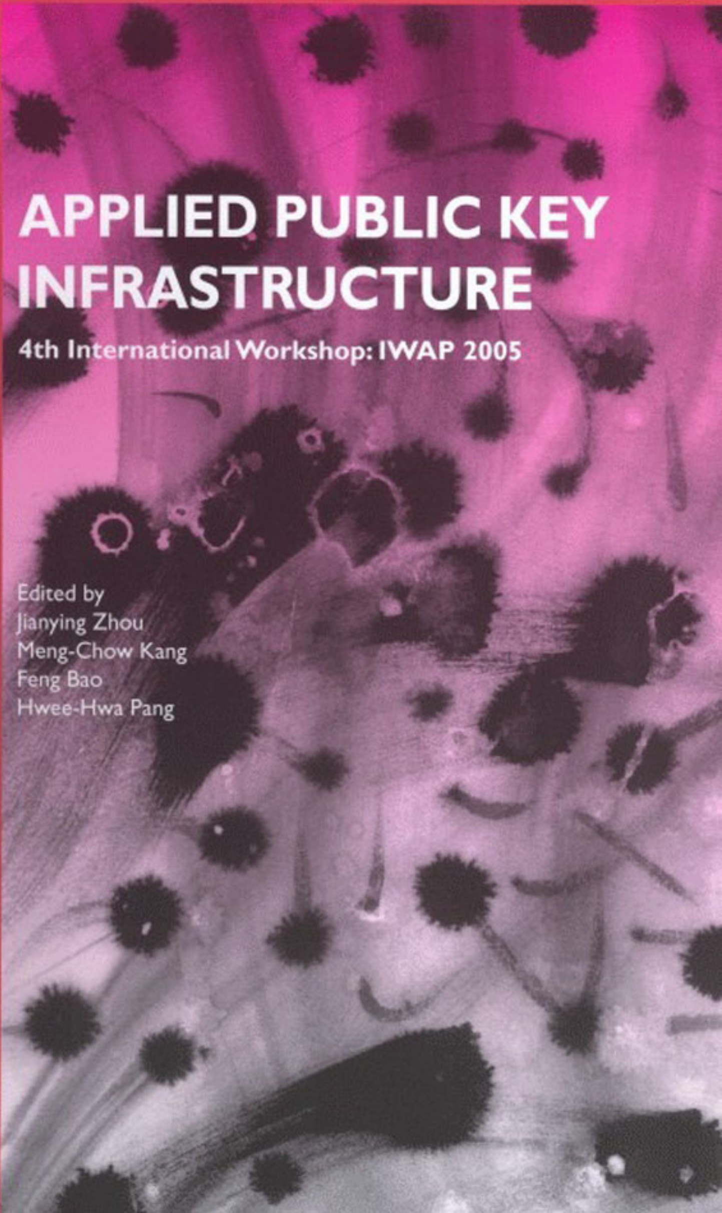
Frontiers
in
Artificial
Intelligence
and
Applications



APPLIED PUBLIC KEY INFRASTRUCTURE

4th International Workshop: IWAP 2005

Edited by
Jianying Zhou
Meng-Chow Kang
Feng Bao
Hwee-Hwa Pang

IOS
Press

APPLIED PUBLIC KEY INFRASTRUCTURE

Frontiers in Artificial Intelligence and Applications

Volume 128

ISSN 0922-6389

Applied Public Key Infrastructure

4th International Workshop: IWAP 2005

Edited by

Jianying Zhou

Institute for Infocomm Research, Singapore

Meng-Chow Kang

Microsoft Asia Pacific, Singapore

Feng Bao

Institute for Infocomm Research, Singapore

and

Hwee-Hwa Pang

Singapore Management University, Singapore

IOS Press

Amsterdam • Berlin • Oxford • Tokyo • Washington, DC

ISBN 1-58603-550-9
Library of Congress Control Number: 2005931171

Publisher
IOS Press
Nieuwe Hemweg 6B
1013 BG Amsterdam
Netherlands
fax: +31 20 687 0019
e-mail: order@iospress.nl

Distributor in the UK and Ireland
IOS Press/Lavis Marketing
73 Lime Walk
Headington
Oxford OX3 7AD
England
fax: +44 1865 750079

Distributor in the USA and Canada
IOS Press, Inc.
4502 Rachael Manor Drive
Fairfax, VA 22032
USA
fax: +1 703 323 3668
e-mail: iosbooks@iospress.com

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Preface

Over the past years, Public Key Infrastructure (PKI) technology has evolved and moved from the research laboratories to the mainstream, in which many organizations are now leveraging it as part of their core infrastructure system for providing and building security in their businesses. Understanding the challenges and requirements of PKI related operations through the sharing of case studies are critical to supporting the continued research and development of PKI technologies and related systems and applications to further progress and innovate for enhancing future development and evolution of PKI in the enterprises.

The International Workshop for Applied PKI (IWAP) is an annual workshop that was initiated in 2001 with the objective of focusing on research and application of Public Key Infrastructure. The first IWAP was held in Korea in 2001 with the active contributions and participation of Asian experts who shared valuable experiences on constructing PKI, in particular, on connecting PKI among Asian countries. The second IWAP was held in Taiwan in 2002, providing an opportunity for participants to exhibit and share their experiences in PKI construction. In 2004, Japan hosted the third IWAP workshop, further discussing the trend and issue on PKI-technologies.

The 4th IWAP workshop was held in Singapore on September 21–23, 2005, in conjunction with the 8th Information Security Conference (ISC'05) and the 1st Secure Mobile Ad-Hoc Networks and Sensors Workshop (MADNES'05). This is the first year for IWAP to have the formal proceedings published by IOS Press and available at the workshop. Selected papers in the IWAP'05 proceedings will be invited for submission to a special issue of the Journal of Computer Security.

A total of 43 submissions were received, of which the program committee selected 15 papers from 11 countries for inclusion in the proceedings. In addition, to enrich the workshop program, 3 valuable contributions originally submitted to ISC'05 were introduced into IWAP'05 program with the authors' consent. This workshop consists of one keynote speech and six technical sessions, covering the topics of PKI Operation & Case Study, Non-repudiation, Authorization & Access Control, Authentication & Time-Stamping, Certificate Validation & Revocation, and Cryptographic Applications.

This workshop was made possible only through the contributions from many individuals and organizations. We would like to thank all the authors who submitted papers. We also gratefully acknowledge the members of the Program Committee and the external reviewers, for the time and effort they put into reviewing the submissions.

Special thanks are due to Ying Qiu for managing the web site for paper submission, review and notification. Patricia Loh was kind enough to arrange for the workshop venue, and takes care of the administration in running the workshop.

Last but not least, we are grateful to Institute for Infocomm Research and Microsoft Asia Pacific for sponsoring the workshop.

Jianying Zhou and Meng-Chow Kang – Program Chairs
Feng Bao and Hwee-Hwa Pang – General Chairs

IWAP 2005 The 4th International Workshop for Applied PKI

September 21–23, 2005
Singapore

Organized by
Institute for Infocomm Research, Singapore

Sponsored by
Institute for Infocomm Research, Singapore
and
Microsoft Asia Pacific, Singapore

Sang-Jae Moon (KNU, Korea)
David Naccache (Gemplus, France & RHUL, UK)
Rolf Oppliger (eSECURITY Technologies, Switzerland)
Susan Pancho (Univ. of Philippines, Philippines)
Kenny Paterson (RHUL, UK)
Bart Preneel (K.U.Leuven, Belgium)
Zulfikar Ramzan (DoCoMo Labs, USA)
Reihaneh Safavi-Naini (Univ. of Wollongong, Australia)
Kouichi Sakurai (Kyushu Univ., Japan)
Tsuyoshi Takagi (TU Darmstadt, Germany)
Vijay Varadharajan (Macquarie Univ., Australia)
Guilin Wang (I2R, Singapore)
Duncan Wong (City Univ. of Hong Kong, China)
Shouhuai Xu (UTSA, USA)
Sung-Ming Yen (National Central Univ., Taiwan)
Jukka Ylitalo (Ericsson Research, Finland)
Moti Yung (Columbia Univ., USA)

Organizing Committee

Patricia Loh (Institute for Infocomm Research, Singapore)
Ying Qiu (Institute for Infocomm Research, Singapore)

External Reviewers

Thodoris Balopoulos
Benoit Chevallier-Mames
Jordi Forne
Jun Furukawa
Jie Guo
Don-Guk Han
Thomas Hardjono
John Iliadis
Dimitrios Lekkas
Shiqun Li
Changshe Ma
Philip MacKenzie
Sumio Morioka
Takeshi Okamoto
Sassa Otenko
Katja Schmidt-Samoa
Goran Schultz
SeongHan Shin
Kristian Slavov
Ken'ichi Takahashi
Xiaojian Tian
Yoshifumi Ueshige
Lionel Victor
Yuji Watanabe
Guomin Yang
Robert W. Zhu

Contents

PKI Operation & Case Study

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005
© 2005 The authors. All rights reserved.

PKI Challenges: An Industry Analysis

Geraint Price [1]
Information Security Group
Royal Holloway, University of London

Abstract. In this paper we categorise some of the challenges facing those building, deploying and using Public Key Infrastructures (PKIs). Our work is based upon a series of in-depth interviews and analysis. The aim of the work in this paper is twofold: to present the conclusions drawn from work that is based on years of practical experience of those in the field; to analyse those conclusions in order to highlight research avenues that will answer the challenges raised by those in industry.

Keywords. Case study, Certificate policies, Regulation, Interoperability, Standards

1. Introduction

In this paper we categorise some of the challenges facing those building, deploying and using Public Key Infrastructures (PKIs). Our conclusions are drawn from a significant and in-depth interview process, in which various parties from industry, government and academia discussed their experiences and views on all aspects of PKI development and deployment.

While we use what was in essence a wide-ranging and detailed case study as a starting point, we believe that the main strength of our work is our analysis of how these initial findings should drive the future direction of PKI research and development.

Our aim in this paper is twofold. Firstly, we highlight what we consider to be the important conclusions drawn from the interview process — conclusions that are explicitly based on many years of practical experience. Secondly, we provide a personal analysis of how these conclusions should direct further work by those involved in developing and researching PKIs.

Due to the breadth of topics raised during the interview process, we break down our analysis into technical and non-technical issues. For example, in the technical domain, we discuss the support for Privilege Management Infrastructures (PMIs), and how authorisation using certificate-based credentials can be used to provide cost saving technological benefits. In the non-technical domain,

[1] Correspondence to: Geraint Price, Information Security Group, Mathematics Department, Royal Holloway, Egham, Surrey, TW20 0EX, UK. Tel: +44 (0)1784 414 160; Fax: +44 (0)1784 430 766; E-mail: geraint.price@rhul.ac.uk.

we analyse how the impact of legal stances on digital signatures are affecting deployment, as well as how compliance with standards and regulation is seen by those implementing a PKI within an organisation.

While we believe that the research agendas we propose are important, it is inevitable that we may be unaware of existing work which already deals with some of the issues we raise. In these cases, we believe that there needs to be a concerted effort to bring such work to the attention of those in industry who use and rely upon PKI. It is clear from our discussions during the interview phase that, if there is existing work which deals with their concerns, then the message does not appear to be getting through.

The remainder of this paper is organised as follows. Section 2 provides a brief overview of the nature and structure of the project which delivered the conclusions in this paper. Section 3 covers issues of a technical nature. In Section 4 we discuss issues that are of a non-technical nature. We use Section 5 to introduce certain opinions which were not directly influenced by the conclusions drawn from our original work. Nevertheless, our suggestions in this section are, by necessity, influenced by our analysis during the reporting process. Finally, we draw some overall conclusions from our work in Section 6.

2. Industrial Research Collaboration

In this section, we provide a brief overview to the research project which provides the background to this paper. The project that gave rise to our source document was the PKI Club [1], which was organised and run within the Information Security Group at Royal Holloway, University of London. The club was a diverse collaboration between interested parties from industry, government and academia. The founding principle of the collaboration was to provide a forum by which issues of relevance to the deployment of PKIs could be discussed in an open and frank manner.

At the time that this project was established, the PKI industry was suffering as the result of people scaling back their expectations. Although PKI had been around for many years, it was perceived to have failed to deliver on its early promise.

As a result, the core questions which the PKI Club tried to address were built around a discussion of the issues that impact upon PKI, and how that impact would shape the future of PKI. As well as being a discussion forum in itself, one of our tasks within the project was to produce a report that would be of use to the member organisations. We provide a more detailed overview of this report in the following subsection.

2.1. Background to the Report

The source document, from which we extracted the conclusions to form the starting point for this paper, was the result of a lengthy and detailed succession of interviews which spanned several months in the latter half of 2003. In addition to

[1] `http://www.isg.rhul.ac.uk/research/projects/pkiclub/`

the interviews, there was an intensive follow-up phase in which discussion continued between ourselves and those interviewed based on the content of their interviews. Following this, there was a period of analysis of the interview transcripts, which allowed us to combine the information presented, bringing together the individual views in a structured manner. We built on this analysis through writing a report, inviting those we interviewed to comment on our initial findings. After taking into account this additional feedback, the final edited version of the report was released to the members of the PKI Club in October 2004.

Twelve interviews were conducted, with each, on average, lasting around two hours. Within the interview process, a total of eighteen people were interviewed and, through their experiences, some seventy projects with PKI relevance were discussed.

In relation to geographical region, the majority of the projects were based in the UK or mainland Europe, while a few of those we interviewed had experience of projects in North America.

Those we interviewed represent both users and implementers in a number of diverse industrial sectors. The sectors covered through their experiences were: financial services, insurance, government, military, telecommunications, security service providers, security product developers and consultancies. Thus, while every relevant party within industry might not be represented, we believe that the breadth of the views covered does not detract from the contribution of the original report. One of the great strengths of the process was our use of a relatively broad and loosely structured discussion which allowed those being interviewed to present what *they* saw as important.

Due to the confidential nature of the interviews, the interviewees were extremely frank and open in their communication. The transcripts were then subjected to filtering and anonymisation before selected attributable quotes were released for the original report. However, the rigorous level of anonymisation and aggregation means that almost all points raised during the interview process were preserved. We believe that the nature of our interaction with the interviewees — which was constructed free of any commercial considerations and constraints — led to a very open atmosphere where heartfelt views were expressed which might have otherwise been suppressed.

The result was an extremely in-depth examination of the issues raised, in a report that totalled over 100 pages.

While the original report was only released as a confidential internal document for the use of those within the broader collaborative project, the group members have kindly allowed us to use the anonymised conclusions from our discussions within this paper. Even allowing for such a high degree of anonymisation we believe that it is a reflection of the challenges facing the PKI industry that we have struggled to compress all the issues which we believe should be raised into this paper.

In addition, an extended abstract of the report has been released [11]. This public version of the report combines the executive summary along with a detailed and in-depth summary of the main chapters of the original report. In doing so it provides a more detailed representation of the report conclusions presented here, along with many other interesting results which we were unable to discuss in this paper for reasons of brevity.

2.2. Presentation of our Results

We now describe how we will present our results from the collaborative report, and how we use those conclusions as the basis for our analysis.

To clarify our analysis we present each of the source conclusions in turn, followed by what we consider its impact to be for those in the PKI industry. We categorise these into two types of statements:

Drivers : These are the consequences of the conclusions which we believe should provide a motivation for a change within the industry or within the use of PKI.

Research Agenda : These are the issues which we believe should provide research avenues within the academic and industrial research community.

The requirement for anonymity, along with a limitation of space, means we cannot engage in a full discussion of how we arrived at our original conclusions. However, because of the way in which the conclusions are drawn from the experiences of those in industry, we believe that the conclusions have sufficient weight for the purposes of our analysis here.

3. Report Conclusions: Technical Issues

In this section, we overview the conclusions drawn from the technical issues covered in the original project. We break down the analysis further, with the technical considerations for asymmetric cryptography being covered in section 3.1, and the technical considerations for the infrastructure itself discussed in section 3.2.

3.1. Technical Considerations for Asymmetric Cryptography

We begin our analysis with a description of the conclusions raised surrounding the use of the core technology — public key cryptography.

PKI's ability to support multiple security services, in combination with its ability to provide non-repudiation, gave it a clear competitive advantage in some scenarios. However, there was often a discrepancy between what the technology was originally implemented to help provide, and what businesses ended up using it for. For example, one interviewee noted that although Identrus [9] had been designed to help implement non-repudiation, it was primarily being used to provide authentication.

Research Agenda : We belive that this shows a lack of understanding by those developing the technology about what businesses genuinely need. As such, we believe that there is scope for an in-depth analysis of how the security services supported by public key cryptography can provide clear support for specific business processes, and thus offer a clear business benefit.

When we discussed the use of asymmetric cryptography, an interesting point was made with regard to the management of the private keys and certificates. While a public key can provide a cleaner *one key* view of an individual within a security domain, in some cases this generated additional problems. One example of this was that using the same key and certificate across multiple layers with different risk profiles greatly increased the complexity of the design. It also made managing the risk profiles more difficult.

Research Agenda : This offers a research opportunity related to certificate management. Specifically, there needs to be an improved means of managing a certificate in relation to multiple policies, where different areas within a security domain might have different requirements in their use of the certificate [2].

3.2. Technical Considerations for the Infrastructure

We continue our analysis with a description of the conclusions concerning the technical aspects of the infrastructure.

The fact that PKI has been largely developed in isolation from specific applications was seen as a key reason why integration has been a problem. The fact that early implementations were built by vendors with no specific application or sets of regulations in mind, meant that it was difficult to get the vendor's products to inter-operate.

Additionally, integration with specific vendor applications was considered more important than compliance with individual standards. It is a commonly held view that while standards provide some benefit, they do not provide everything and are often used as a template for education rather than as a strict set of guidelines. It was suggested that one of the reasons for this is that standards are often compromised in order to get things to work in practice.

Driver : This clearly provides a driver for improved infrastructure support for specific business applications. While there have been past initiatives to improve integration, their impact has been limited. In addition, while standards have been lauded for adding to the pool of information, the discussion above demonstrates that they are not the silver bullet that some would hope for.

Although supporting legacy applications still causes some problems, the use of middleware tools has reduced the potential headache to a manageable level. Also, one interviewee noted that when deciding where to position the interface to the PKI, if the PKI was being used to support multiple applications, then it was of benefit if it was mediated through some form of middleware component. Only if the PKI was being used exclusively to support a single application did they consider it wise to interface directly to the PKI.

[2] As we see from section 4.2, certificate policy management in itself is something that requires further work and understanding.

Research Agenda : The fact that middleware is being used in these areas in practice, suggests that further research into middleware development could help improve certificate support in environments with diverse security requirements.

When a security service is being migrated either from an old PKI to a new PKI, or from some other security technology to a PKI, knock-on effects are felt. Care needs to be taken when considering any modifications that are carried out to the original infrastructure. For example, the objective of one project was to perform an upgrade of the PKI vendor technology. The upgrade process had to take into account changes which had been made to the original product, where these changes were not supported in the conventional upgrade package. This complicated the project significantly.

Research Agenda : We believe problems such as these, when encountered even within a specific vendor's products, add weight to the argument we make above for research into the use of a middleware component.

Integrating with, and providing support for, a Privilege Management Infrastructure (PMI) is considered to be another important issue. A few people noted that PMIs have the ability to provide a more tangible business benefit in the form of long term savings through cost reduction. However, many people felt that the business decisions needed to support a PMI were difficult to outsource. This could potentially have an impact on those companies offering PKI as a hosted service, as integration across an organisational boundary is generally more difficult.

Driver : Many people we talked to noted the difficulty in providing a sound business case for implementing a PKI. PKI's ability to support other security infrastructures (e.g. PMIs), that independently offer a valid business proposition, should provide an avenue for the PKI industry to exploit.

Research Agenda : It would be an interesting research question to consider whether it is possible to either: develop the technology to make outsourcing of a PMI less difficult; or to continue to outsource the PKI while providing better inter-organisational support for the PMI.

A few people noted that a central PKI had been built even when disparate parts of the organisation needed separate functionality. One of the reasons given for re-using an infrastructure was to centralise the control of various PKI projects in a large organisation. Interestingly, re-use is not always possible, with some implementations being built as a *black box*, whereas others fall foul of scheme or legal requirements.

Research Agenda : We believe that a better understanding of where and when infrastructure re-use is possible, or even desirable, is required. Understanding these issues should improve the ability to propose a sound business proposition on a case-by-case basis.

4. Report Conclusions: Non-technical Issues

In this section, we review the conclusions which we draw from the non-technical issues covered in the original project. We break down the analysis according to legal and regulatory considerations (section 4.1), management of the infrastructure (section 4.2), commercial considerations (section 4.3), and the human impact (section 4.4).

4.1. Legal and Regulatory Considerations

We begin our review of the non-technical impact areas by looking at the legal and regulatory factors that can have an affect on a PKI deployment.

When we reviewed the legislation drawn up to support the use of digital signatures, many cited the continuing uncertainty surrounding the implementation of the legislation as a problem. Another concern raised was the notion of control over the private key — as discussed in the EU Directive on Electronic Signatures [7] — and how to ensure that an implementation conformed to this part of the legislation. Such uncertainty has forced many of those implementing PKIs to rely upon contract law rather than on statutory law when ensuring that signatures will be honoured.

Driver : Currently, asymmetric cryptography is the only technology capable of meeting the criteria for *Advanced Electronic Signatures* set down in the EU Directive on Electronic Signatures. Thus, it strikes us that the problems generated by the legal uncertainties are specific to those within the PKI industry. As a result, it is in the interest of those in the PKI industry to see that these issues are resolved.

Research Agenda : There are numerous hardware products that can be used to protect private keys [3]. However, we believe that there is a possible research avenue based on demonstrating how such devices can be used in a deployed system in a manner that clearly adheres to the notion of control outlined in law [4].

Compliance with specific industry regulation was seen by many as an important driving factor. Regulation was seen as an aid to developing realistic policies and procedures for implementing PKIs. Identrus [9] was isolated as a useful tool to follow for those building an implementation.

Driver : As we note in section 3.2, implementations have often been hampered by the lack of a target application. We believe that following regulation for specific industries can provide a useful tool for ensuring that implementations are successful.

[3] For example, hardware security modules (e.g. `www.ncipher.com`), or smartcards (e.g. `www.gemplus.com`).

[4] Such research could potentially focus on the procedural, technical or computing process elements of the key management life-cycle.

Liability was seen as a concern for those implementing a PKI. Uncertainty as to where the liability lies has been responsible for driving up the cost in many implementations. Finding ways of limiting liability was seen by some as a large part of the design and process cost.

Research Agenda : We believe there is scope here for a technical solution which could help reduce the costs that liability brings. An example of the type of technology we see fulfilling this role might be similar to those where cryptographic mechanisms are implemented in order to maintain anonymity (e.g. Chaum's work on digital pseudonyms [4]). Another example of the type of techniques which could be used are those where there is shared generation of cryptographic keys, where the trusted authority does not have access to the final working key (e.g. the work carried out by Al-Riyami and Paterson [1]).

4.2. Management and Administration of the Infrastructure

We now consider the effect that the management and administration of the infrastructure has on the use of the cryptography.

In discussing revocation, many of those interviewed voiced preferences for and against the Certificate Revocation Lists (CRL) [8] or On-Line Certificate Status Protocol (OCSP) [10]. One interesting point raised was how the decision on whether to use a CRL or OCSP could be tied to the time criticality of the action being supported. In some cases the PKI only provides the front end authentication and there are separate mechanisms to lock people out of the authorisation modules on individual applications. In such cases, where there are other means of revoking the right to carry out any time critical actions, CRLs were deemed sufficient for the authentication process.

Research Agenda : We believe that there is a prime research opportunity here into how such "secondary" revocation might improve a PKIs ability to withstand compromise, without requiring the additional burden of an online service.

When we discussed the management of system security from a commercial perspective, it was considered important to realise that PKI policies and procedures exist in the context of a larger corporate security policy. This can be an important consideration when analysing how a corporate security policy might influence the PKI security policy and vice versa.

Research Agenda : We believe that there is further scope for research into how PKI policies interact with, and are influenced by, corporate policies. This is likely to be very interesting in areas where corporate policy is often shaped by industry or government regulation.

The logical and physical management of the infrastructure can, when necessary, be separated to aid those with different and possibly competing interests. A good example of this is the Identrus infrastructure which can be split into three parts: certificate management; user registration; application. These can then be run by separate entities.

Research Agenda : We believe this is an ideal opportunity to conduct research into gaining a better understanding of the types of business processes which can be segregated in this way. This could then allow parties with possibly conflicting interests to manage their own segment [5].

Certificate Policies (CP) and Certification Practice Statements (CPS) provided a large amount of discussion during the interview process. Clarifying the content of each was problematic for some of those we spoke to. Many felt that the confusion surrounding them was one of the biggest problems they faced. One interviewee felt that in a contractual infrastructure, using a CP and CPS is vitally important for ensuring the legal responsibilities of each party are more easily identified. The *de facto* reference for building a CP and CPS — Internet RFC 2527 [5] [6] — was deemed by many to be insufficient for their requirements when they come to writing their own.

Driver : The impact that a CP and CPS have on the legal aspect of managing a PKI will ensure that they will continue to play an important role in any major implementation. Therefore, if we are to bring down the cost of their development, it is crucial that the uncertainties and difficulties are addressed over time.

Research Agenda : We believe that specific case studies need to be openly identified and examined. The output of such a project should be the provision of more targeted example documents. This should aid those building a CP or CPS with little or no prior experience. In addition it should help identify the key areas where they are most useful in practice.

4.3. Commercial Considerations

We now provide a summary of the commercial concerns raised.

When we asked the interviewees where PKIs had been deployed, one of the most common responses was that it was used to protect "high value transactions". Given the additional security achieved, and cost incurred, then supporting such high value transactions is going to be a natural place for deployment of a PKI. However, it is also our view that considering the value of the transaction alone does not provide us with sufficient information when we consider what types of applications PKI might best support in the future.

Research Agenda : We believe that there needs to be a more complete understanding of the nature of the transactions that PKI supports best. What are the types of communicating party relationships? What are the process flow models? What type of contractual environments provide the best scope for addressing the process issues? Answering these, and similar questions, will provide the industry with a clearer understanding of where PKIs can best be used in the future.

[5]We believe this ties in well with our discussion in section 4.1 on minimising liability.
[6]Although RFC 2527 has now by superceded by RFC 3647 [6].

One of the main business decisions facing those considering implementing a PKI is whether to build the infrastructure in-house, or use a hosted service. For organisations looking to implement a PKI to secure high value transactions, the issue of retaining control over the whole infrastructure could influence this decision. PKIs are often used to support applications and PMIs, but those secondary services are generally not as easily outsourced to a hosted service. This means that there are increased integration and business process issues to tackle if the PKI is off-site while the PMI is on-site.

Research Agenda : We believe the commercial considerations voiced here add significant weight to our argument in section 3.2 for researching the ability for PKI to more adequately support a PMI across a domain or organisational boundary.

Although PKIs are more likely to be of use within large organisations, their deployment within such organisations is likely to bring its own set of problems. In the context of an organisation formed from a diffuse group of companies, the sector responsible for the group IT function can sometimes lack the authority necessary to direct the project within the separate companies. This can lead to greater interoperability problems.

Research Agenda : We believe that this supports our call in section 3.2 for further research into the use of middleware components. A centralised PKI which provided a middleware managed interface could ease the pain of interoperability concerns in a diffuse organisational structure.

4.4. The Human Element

Most definitions of what a PKI is will acknowledge the role of the people and processes within the infrastructure. In this section, we isolate the issues of relevance to the impact of the human understanding of the technology.

Criticism was levelled at the expectations some people were placing on software key storage, which undermines some of the benefits gained from using asymmetric cryptography. In this case, there is a requirement for increased understanding by those implementing a PKI of the benefits and drawbacks of the various technological choices faced.

Research Agenda : We believe that there is scope for research which highlights the comparative benefits and drawbacks of the technical choices faced. We envisage a project which provides an analysis of the various technical choices and their impact upon the security of the system as a whole. The results would be beneficial for those considering the impact of a particular technological choice.

The end-user's ability to use the technology, along with their acceptance of it, was considered an important concern. In terms of ease-of-use, trying to get the user interface to mimic existing technology of a similar type is seen as one way of improving usability. It is considered important that the user is not hampered when using the security, otherwise they might find ways to bypass it. As well as reducing security, unusable products can reduce usage and uptake of new applications. In some scenarios this has led to new applications being dropped because they were too unwieldy, even after a lot of money had been spent on their implementation. Many felt that the usability of early PKI products had suffered because the user interface had been designed by those responsible for the underlying cryptographic technology.

Research Agenda : We believe that more needs to be done to understand the usability impact of public key cryptography. Security experts take for granted a lot of the terminology and understanding of the underlying technology. However, we are not the experts in usability studies. Although there is some recent work on understanding the Human Computer Interaction (HCI) aspects of security [7], we believe that more specific work is required to understand the particular problems faced when using public key cryptography.

5. Extended Analysis

This section highlights potential research avenues which are not directly attributable to the conclusions of the original report, but are our own views which were heavily influenced by our analysis during the reporting process.

When PKI entered the marketplace, it was sold on the basis of the underlying cryptography. As it became apparent that this did not result in effective use of the technology, those in the industry focused on selling the services provided by the technology (e.g. authentication, signatures, etc.). While this move is to be lauded, we believe this progression needs to go one step further, with PKI vendors selling those security services in terms of the types of application each service best supports. For example, one clear use of an authentication service is in support of a PMI.

Research Agenda : If this move is to be successful, there needs to be further research that clearly identifies the application processes which each security service best supports.

We believe that in designing the infrastructure to support the technology, there has been too much focus to date on the subscriber-CA and CA-CA relationships and not enough on the subscriber-relying party and relying party-CA relationships. Such issues are being brought to light with the implementation of more 3-party models such as those used in BACSTEL-IP [2].

Research Agenda : While such developments are already taking place in practice, we belive that fundamental research into these relationships could aid our understanding of their impact.

[7]For example, recent work by Sasse [3,13]

When we discussed the use of PKIs to support applications that were external to the organisation that ran the PKI, it was clear that the policies and procedures were much more difficult to manage. This has a knock-on effect upon how secure the resulting use of the private key can be considered to be. A few implementations which we discussed during the course of the interviews limited the types of business processes which can be accessed by externally managed keys.

Research Agenda : While this technique of limiting the available services is a relatively straightforward design decision, we believe that there is scope here for developing a firmer understanding of this type of risk management and policy management.

The cost of deploying a PKI is undeniably large and any significant saving can increase the likelihood of a project providing a financial business benefit.

Research Agenda : While the above statement is easy to make, finding solutions to the problem is far from straightforward. One example of how we believe this might be achieved is through developing more lightweight infrastructures. By targeting specific business processes and building the technology directly into targeted applications, we believe there is clear scope for making the implementations more lightweight in the future [8]. It would be interesting to see what other forms of cost reduction are also possible.

As we discussed in section 4.1, many of those we spoke to voiced concern over the issue of liability within PKIs. It would appear to us that well-defined schemes have the potential to allow for greater control over such liabilities. By signing up to a specific set of membership or scheme rules, all parties should be able to make use of a PKI with a clearer understanding of the risks they are undertaking.

Driver : The concern over liability, coupled with the view we express above on the benefit of memberships or schemes, appears to provide yet another driver towards more targeted deployment of PKIs in support of specific business processes.

When making a decision on which security technology to use, it is important that those making the decision are able to clearly define the inherent risks being faced along with their potential impact. It would appear to us that the ability to tie the cost of implementation to both business benefit and associated risk is still in its infancy. We believe this to be especially true for the risk analysis aspect of this equation. This view is not only our own, as we have attended presentations in the past where conventional risk management was deemed inadequate for IT security [12].

Research Agenda : We believe that improving our understanding of the interaction between risk analysis and security infrastructure could provide a fruit-

[8]Examples of such lightweight targeted use of asymmetric cryptography were given during our research. However, we believe that the possibility of doing so has only limited acceptance at present.

ful area of research. This is likely to be more important for designers of public key systems, due to the more "open" nature of the security associations afforded by the underlying technology.

6. Conclusions

This paper represents an insight into crucial issues facing those designing, building and using PKIs. The conclusions of our source project are the opinions of those who matter — designers and users of PKI technology in practice.

In addition to the thoughts of others, we present a number of research directions that we believe need to be explored in more detail. We believe that the points we raise — when combined with the source opinions for others to analyse — provide a useful tool for those in both academic and industrial research environments to develop solutions to these problems.

We close our discussion by highlighting what we see as the most important overall conclusions presented.

A global theme that is present in our work is that we believe there needs to be a better understanding of how a PKI can be targeted to support specific business areas or types of business process models. For example, many people identified their disillusionment with RFC 2527 [5] [9] as a template for developing CPs and CPSs. More specific examples need to be developed and analysed. From this we can arrive at a better understanding of the types of processes which can be supported by specific categories of security policies.

In addition, research needs to be conducted to provide a clearer model of which business processes, and hence benefits, are directly supported by specific classes of security services.

For example, providing strong authentication is seen as the main benefit to be gained from using a PKI. Supporting authorisation mechanisms is the natural place to use such authentication. Thus, there needs to be greater promotion and understanding of what types of authorisation processes are best suited to asymmetric cryptography. Another example is the use of asymmetric cryptography to provide integrity services. A few of the implementations that were discussed used the strength of this opaque integrity mechanism (where many signatures can be concurrently added to the same document). Where else might they be deployed?

Understanding the scope and limitations of these benefits will allow us to build more useful technology and help deliver more tangible business benefits. Although the conclusions to such research projects are of most use to the business community, we believe it is important that we in the research community listen to their requirements when designing new solutions.

Acknowledgements

The work carried out in the PKI Club at Royal Holloway formed the starting point for our work. Fred Piper and Kenneth Paterson were responsible for contributing

[9] Although RFC 2527 has been superceded by RFC 3647 [6]

at both a technical and organisational level to the running of the PKI Club. We would like to thank them for their contribution to the platform of work on which this paper is built. We would also like to thank those PKI Club members who gave their time graciously during both the interview stages and the follow up discussions. In addition, we would like to thank Chris Mitchell, Julia Horn and various anonymous referees for improving on the presentation of this paper.

References

[1] S.S. Al-Riyami and K.G. Paterson. Authenticated three party key agreement protocols from pairings. Cryptology ePrint Archive, Report 2002/035, 2002. `http://eprint.iacr.org/`.

[2] `http://www.bacs.co.uk/bpsl/bacstelip`.

[3] S. Brostoff and M. A. Sasse. "Ten strikes and you're out": Increasing the number of login attempts can improve password usability. Paper presented at the CHI 2003 Workshop on Human-Computer Interaction and Security Systems, Ft. Lauderdale, April 2003.

[4] D. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, February 1981.

[5] S. Chokani and W. Ford. RFC 2527: Internet x.509 public key infrastructure certificate policy and certification practices framework, March 1999.

[6] S. Chokhani, W. Ford, R. Sabett, C. Merrill, and S. Wu. RFC 3647: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework, November 2003. Obsoletes RFC 2527.

[7] EU Directive 1999/93/EC of the European Parliament and of the Council on a Community framework for electronic signatures, December 1999. `http://europa.eu.int/eur-lex/pri/en/oj/dat/2000/l_013/l_01320000119en00%120020.pdf`.

[8] R. Housley, W. Polk, W. Ford, and D. Solo. RFC 3280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, April 2002. Obsoletes RFC 2459.

[9] `http://www.identrus.com`.

[10] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams. RFC 2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP, June 1999.

[11] G. Price. PKI – An Insider's View (Extended Abstract). Technical Report RHUL–MA–2005–8, Department of Mathematics, Royal Holloway, University of London, June 2005. Available from: `http://www.rhul.ac.uk/mathematics/techreports`.

[12] R. Walton. Uses and abuses of cryptography (invited paper). In Kenneth G. Paterson, editor, *Proceedings of the* 9th *IMA International Conference on Cryptography and Coding*, volume 2898 of *Lecture Notes in Computer Science*, pages 125–132. Springer-Verlag, 2003.

[13] D. Weirich and M. A. Sasse. Pretty Good Persuasion: A first step towards effective password security for the Real World. In *Proceedings of the New Security Paradigms Workshop 2001*, pages 137–143. ACM Press, 2001.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Directory Based Registration in Public Key Infrastructures

M. Lippert, E. Karatsiolis, A. Wiesmaier, J. Buchmann
Technische Universität Darmstadt, Fachbereich Informatik
Hochschulstr. 10, 64289 Darmstadt, Germany
{mal, wiesmaie, karatsio, buchmann}@cdc.informatik.tu-darmstadt.de

Abstract. Certificate life-cycle management operations, especially the initial registration of users, are important but expensive tasks in Public Key Infrastructures. In this paper we propose a mechanism that integrates these tasks with established processes, technology and data. Our approach is based on directory services that are usually already employed by companies, public administration and similar organizations for non-PKI purposes. It spares additional personell and is less error-prone, since the utilized processes are already set up and known to administrative personell and users. This reduces the costs to bootstrap and operate a Public Key Infrastructure. We show a case study about the proposed mechanism that was conducted at the Technische Universität Darmstadt in Germany in order to supply 20,000 students with certificates and keys.

Keywords. LDAP, Trustcenter, PKI, Registration, X.509

1. Introduction

Public Key Infrastructure (PKI) is an effective technology for securing digital communication in large networks like an organization's intranet or the internet. It provides the services of authentication and non-repudiation as well as confidentiality. Many applications already rely on these services like secure e-mail based on S/MIME, secure access to web sites using SSL/TLS, software updates in modern operating systems, and home banking.

Many companies and similar organizations like universities already employ directory services for a variety of non-PKI purposes. These are for example electronic address books, support of network-wide access control and maintainance of information relevant for network-administration. They usually contain data about employees, computers and digital services within the organization's environment. Moreover, the procedures how the data gets into the directory as well as their life-cycle management are already established and well known to both administrators and users. These procedures are already set up in a way that satisfies the organization's needs for security and usability. The directory can be assumed to be complete and up-to-date. Users are able to authenticate to the directory. The directories usually provide an interface that conforms to the Lightweight Directory Access Protocol (LDAP).

PKIs employ directory services as well. In this context they are used for publishing certificates and certificate revocation information. In [1] we have utilized LDAP directories to support two important PKI functions. One is to provide a proof-of-possession for keys that can be used only for encryption. This is done by placing the certificate encrypted on the directory and only the legitimate user (the one who possesses the private key) can decrypt it and place it in the directory. This task is associated to the registration of an entity in the PKI. The other is the delivery of software personal security environments. These are usually encoded according to PKCS#12 [2]. This is achieved with proper use of the access control mechanisms that are available in the LDAP directories. The paper at hand goes one step further. The whole certificate life-cycle management processes are based on a directory. The proof-of-possession scheme is one of the functions needed to implement the registration. We did not employ the delivery scheme since in our PKI we use smart cards. We therefore show how to support a hardware token based PKI.

In [3] a case study is provided where the registration is supported by an Active Directory Server. In contrast to our approach, this server is specially set up for this purpose. It is supplied with user information from other directory servers that were present in advance. Other certificate life-cycle management functions are not discussed.

Gutman describes in [4] the PKIBoot protocol which is in many respects similar to our approach. The protocol is intended to bootstrap network components and services in a PKI similar to the concept of Domain Name Services (DNS). In contrast to our approach it relays on the Certificate Management Protocol (CMP) [5] which is rarely used and comes with the flaws described in [4]. Instead our approach is based on LDAP. This is straight forward as common PKI enabled applications already implement the LDAP protocol. They have to be adapted only slightly to use the proposed mechanism.

In this paper we describe the mechanism to utilize existing LDAP directories in order to bootstrap and operate a PKI efficiently and automatically. This saves time and money. The basic idea is to reuse information that is already available in the directory for the registration. Furthermore, it employs the directory as a means to formulate requests for certificate life-cycle operations. These are initial certificate requests including registration, certificate revocation requests, and certificate renewal requests. Utilizing the LDAP's access control mechanisms allows for sophisticated PKI-management strategies as e.g. distributed administrations and group responsibilities. As directories are accessible from all over the network, this also applies for the PKI management. The progress of each operation is reflected by the LDAP and thus visible to trust center operators as well as to entities. The mechanism can further be combined with the proof-of-possesion described in [1].
The paper is organized as follows:

Section 2 gives a short introduction to LDAP directories. It provides details about their common usage and structure. It shows how their structure can be altered dynamically to adapt to new requirements. This is a basic feature for our approach. Section 3 identifies the functions of the Registration Authority (RA) that will be covered by our mechanism. In Section 4 we describe the initial scenario and assumptions. Section 5 defines the new mechanisms. We show how to bootstrap and maintain the PKI in the described scenario. We therefore show how the certificate life-cycle operations are supported by the proposed mechanism. It is divided into three subsections, one per each request (certification, certificate revocation and certificate renewal). The security of our approach is discussed in Section 6. Section 7 shows a case study of the approach conducted at the Technische Universität Darmstadt. In Section 8 we conclude our work and give an outlook on further developments.

2. LDAP Directories

LDAP directories are a common means to organize all objects in a network. This includes hardware components (e.g. computers, network switches, etc.), services (e.g. web servers, virtual private networks) as well as the users. All these objects are modelled as entries of a hierarchically organized Directory Information Tree (DIT). Each entry can be uniquely described by the path leading from it to the tree's root. This path is called the entry's Distinguished Name (DN).

Each entry consists of a set of attributes. In LDAP directories[1] the type of an entry is described by special attributes called `objectClass`. Object classes define which attributes the entry is able to hold, their type and whether they are mandatory or optional. Object classes may be structural, abstract or auxiliary. Structural object classes define the principal nature of an entry. Abstract object classes are meant for defining common aspects of different structural classes. This is done using the concept of inheritance. Abstract classes are not allowed to be used in an entry without an inheriting structural class. Auxiliary classes specialize an entry by adding additional attributes. An entry can take zero or more auxiliary classes. The set of supported object classes define the directory's schema. Directories should always be operated with schema checking turned on. This enforces that the content of each entry always complies to the defined structure.

Directories already provide various security features. These are server authentication, confidentiality of the transmitted data, user authentication, and access control. The first two features are achieved by having the communication wrapped into an SSL/TLS protocol. Thus, they are based on PKI technology. Users authenticate themselves by means of passwords that are related to their directory entries. This is referred to as *binding* to an entry. The passphrase can hereby either travel in clear (Simple Authentication) or as a salted and iteratively computed hash value using Simple Authentication and Security Layer [7]. In the usual setup, directories use cleartext passwords over an SSL/TLS secured communication path. Access control is provided by means of access control lists (ACLs). They assign access permissions to read, modify, create or delete entries and/or attributes. The ACLs distinguish between anonymous and authenticated users.

[1]More specificly, we refer to LDAP version 3 [6]. This is the recommended version to use with PKIs.

Table 1. Object classes commonly used for modelling persons in LDAP directories. (S) denotes that the class is structural, (A) that it is auxiliary.

Object Class	**Description**
`country` (S)	Describes a country.
`organization` (S)	Describes an organization.
`organizationalUnit` (S)	Describes an organizational unit like departments or working groups.
`dcObject` (A)	Adds the concept of domain components for creating entries based on domain names.
`person` (S)	Describes a person (e.g. name, phone number, userPassword).
`organizationalPerson` (S)	Derived from `person`, adds information as postal address and room number.
`inetOrgPerson` (S)	Derived from `organizationalPerson`, adds information usable in the internet like emailAddress, optionally allows certificates.
`pkiUser` (A)	Optionally adds certificates to an entry.
`strongAuthenticationUser` (A)	Mandates a certificate, necessary for PKI based binding to the directory.
`posixAccount` (A)	Describes a UNIX account (UserID, password hash). Used for LDAP based login.

For authenticated users it is further differentiated between actions that concern their own entry[2] or those that concern other entries.

The following fact is important for our approach. The type of an entry can be changed dynamically. This is done by adding new, commonly auxiliary, object classes. If the attributes of an added object class are either optional or reasonable default values exist, the schema change can be done at once before their first usage. Otherwise the type of an entry has to be changed on the fly, when the respective information becomes available. This would require additional checking during regular operations, but still is feasible.

Table 1 lists object classes that are commonly used for modelling persons in an LDAP directory. The first three classes may be used to make up the path in the DIT to the user entries. They usually mirror the structure of the organization and/or the driven applications. The fourth class adds information that can be used to build internet domain names. This is often used in LDAP for the root entry of the tree. The class `person` models a person. It can be specialized to `organizationalPerson` and further to `inetOrgPerson`. The last one optionally takes a certificate for the user. This can also be achieved by adding the auxiliary class `pkiUser` (optional certificate) or `strongAuthenticationUser` (mandatory certificate), respectively. The class `strongAuthenticationUser` provides strong authentication when binding to the entry. The class `posixAccount` allows to utilize LDAP directories for computer login.

[2]The one they did bind to.

3. Certificate Life-cycle Management

Registration is the first step for an entity to become member of a PKI. It is usually the most expensive one. This is because no trust relationship exists between the entities and the trust center prior to their registration. Thus, the authenticity of the registered data must be achieved by out-of-band (mostly manual) means. Registration should meet the following requirements:

1. Identification of the requesting entity.
2. Establishment of data that describes the entity.
3. Enforcement of the certification policy by
   (a) verifying that the entity is allowed to receive a certificate, and
   (b) having the entity to accept the PKI's policy.
4. Generation of a unique distinguished name to be used in the certificate.
5. Linking the established registration data to the certificates produced for the entity. This is necessary to
   (a) detect naming collisions,
   (b) be able to resolve a certificate to the identity of its owner, and
   (c) monitor which certificates belong to which entity.[3]

To be able to undoubtly link certificates to their owners, each operation that modifies the certificate status must be visible to the RA. This can be achieved by having it act as an interface where entities request all certificate life-cycle operations. The RA then passes appropriate requests to a Certification Authority (CA).

For the remainder we assume the following structure of the trust center: It consists of a RA, a Certificate Management Authority (CMA) and a CA. The RA acts as in interface for the entities. All certificate life-cycle operations can be requested there. The CMA is responsible for the postprocessing of the requests. This is mainly the dissemination and publication of certificates and certificate revocation lists. The CA is encapsulated by the two others. It is therefore transparent to the entities. This makes it easier to protect it, since it only communicates with known entities and may be kept offline.

4. Initial Assumption

We now state the assumptions our approach relys on. They have been motivated in the introduction.

For the following we assume that an LDAP directory is already set up in an organization's intranet. It serves as an address book and for computer login. It contains one entry per employee. Each entry holds sufficient data to describe the identity of the respective employee. The entries have the object classes `inetOrgPerson` and `posixAccount`.

[3]In cases where the distinguished name in the certificate is not derived from the entity's identity, this is not trivial. Such certificates are called pseudonymous certificates.

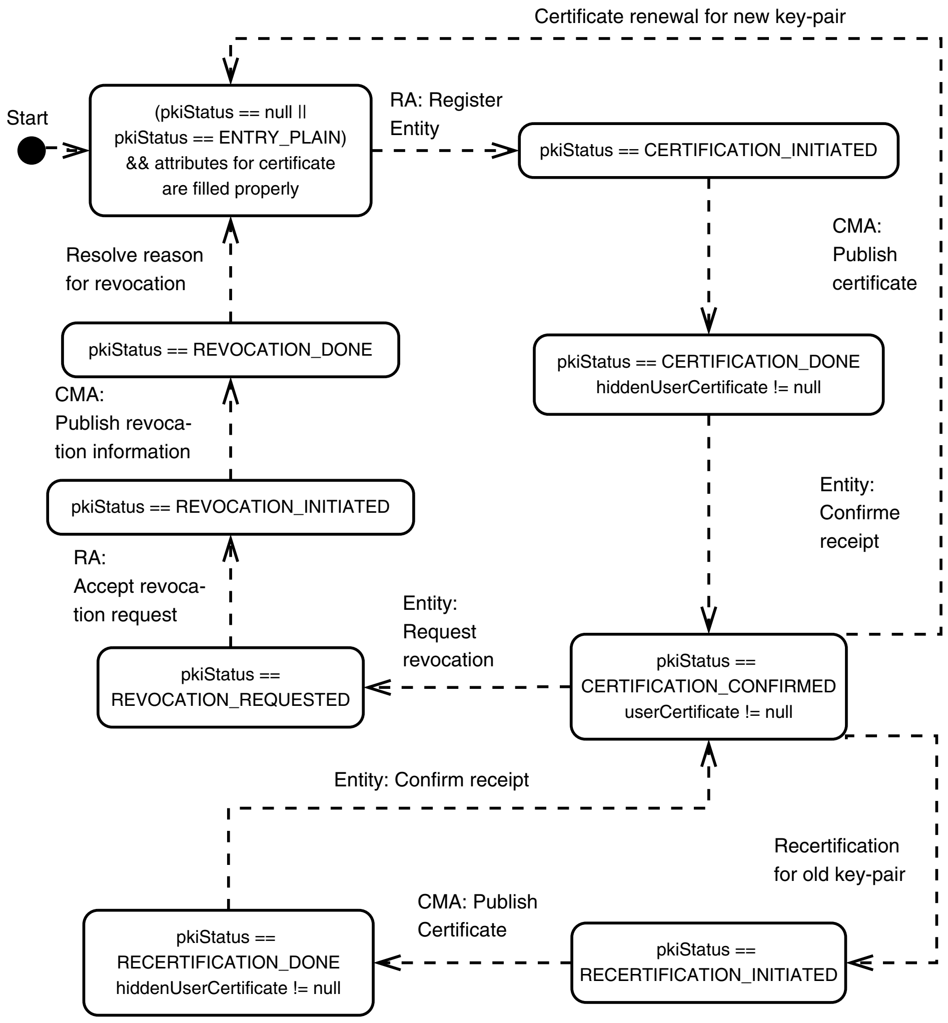

Certificate renewal for new key-pair
Start
(pkiStatus == null || pkiStatus == ENTRY_PLAIN) && attributes for certificate are filled properly
RA: Register Entity
pkiStatus == CERTIFICATION_INITIATED
CMA: Publish certificate
Resolve reason for revocation
pkiStatus == REVOCATION_DONE
pkiStatus == CERTIFICATION_DONE hiddenUserCertificate != null
CMA: Publish revoca-tion information
pkiStatus == REVOCATION_INITIATED
Entity: Confirme receipt
RA: Accept revoca-tion request
Entity: Request revocation
pkiStatus == REVOCATION_REQUESTED
pkiStatus == CERTIFICATION_CONFIRMED userCertificate != null
Entity: Confirm receipt
Recertification for old key-pair
CMA: Publish Certificate
pkiStatus == RECERTIFICATION_DONE hiddenUserCertificate != null
pkiStatus == RECERTIFICATION_INITIATED

Figure 1. Statemachine of the Registration/Certification Process.

The processes that establish and maintain the directory contents are already available and sufficient with respect to the organization's security needs.

Furthermore, we assume that smart cards are employed for storing the employee's private keys and the respective certificates. The cards are pre-personalized. I.e. each card already contains a key-pair, but no user information has been put on it, yet.

5. Mechanism

We now describe the mechanism to reuse the available information and to automate the functions of the RA.

The mechanism can be described as a finite state machine. The state machine is depicted in Figure 1. The state is kept per entity in the directory. For this reason we introduce the new optional integer attribute `pkiStatus`. It is part of the auxiliary object class `pkiManagement`[4] which can easily be added to a directory. The attribute is of type integer to make comparison less costly and thus allow fast search operations. For this reason, it should be indexed by the directory as well. The RA scans the directory in regular intervals and reacts on the status of the respective entries. Also the entities' client software reacts on status changes of this attribute. In contrast to the RA, the client software only has to check the entity's own entry, not the whole tree.

5.1. Requests for Certification

The first operation for each entity is the registration. It spans the process from the identification of the entity to the initial certificate being issued. It is triggered for an entity, if the respective entry is in the following state:

1. The attribute `pkiStatus` does not exist or contains the value `ENTRY_PLAIN`.
2. All attributes that are used for certificate creation are set. This may include
   - the common names ('CN'),
   - the e-mail address ('mail'),
   - the organizational unit ('OU'), and
   - the user-ID

Thus, the requirements 1, 2, 3a, and 3b of Section 3 are achieved by the processes that lead to this initial state. We have assumed them to be sufficiently secure and already established.

The first condition offers a choice. The established processes may be altered to set the newly introduced `pkiStatus` to `ENTRY_PLAIN`. This allows faster scanning of the directory by the RA because the queries become simpler. On the other hand, processes need to be adjusted what may sometimes be unwanted or even impossible.

The RA now generates a certification request for the CA. This request consists of the to-be-signed part of an X.509 certificate (see [8]) and optional operational parameters. Each attribute of the certificate is either derived from the entry's attributes, from constant values defined by the certification policy or from a mixture of both. We take the entry's distinguished name as a meta attribute to be able to use it in requests. After posting the request to the CA, the RA sets `pkiStatus` to `CERTIFICATION_INITIATED`. This indicates that the certification request is on the way. The described mapping achieves the requirements 4 and 5 of Section 3.

We give an example for a mapping from an entry to a to-be-signed certificate. John Doe is a student. He is registered for the department of Computer Science. His directory entry is depicted in Figure 2. It contains his full name 'John Doe' (`CN`), his network account id 'jdoe' (`uid`), the name of the department 'Computer

[4]The definitions of the object class and its attributes are given in the appendix.

```
dn: CN=John Doe, OU=CS, ...
givenName: John
sn: Doe
cn: John Doe
ou: Computer Science
uid: jdoe
mail: jdoe@cs.tu-darmstadt.de
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
```

Figure 2. Example LDAP entry for John Doe

Science' (`OU`), and his e-mail address 'jdoe@cs.university.de' (`mail`). We assume the distinguished name (DN) of the object to be `CN=jdoe, OU=CS, OU=student, OU=addresses`. The RA now generates a certification request based on these values. It is depicted in Figure 3. The `CN` and `OU` attributes are copied to the respective components of the certificate's subject name. The information that John Doe is a student is taken from the entry's DN (`OU=student`). This affects the certificate's subject DN as well as the certificate policy set in the respective extension. For better readability the attribute `OU=student` is inserted right after the `CN` in the subject name. The suffix of the DN `O=university, C=DE` is provided as a constant as demanded by the policy. The attribute `notBefore` is set to the current time, `notAfter` is set to the end of the semester term which is also supplied by the policy as a constant. The e-mail address is added as `rfc822Name` to the `subjectAlternativeName`. Also the entry's DN is added as `directoryName` to the `subjectAlternativeName` to provide a strict binding between the certificate and the entry. This is important for the security of the revocation and further discussed in Section 6.

Especially the mappings that make up the certificate's distinguished name have to be chosen with care. It must be assured that the result is a unique name within the PKI. Of course, the RA is able to detect name collissions since it can distinguish entities by their entry's distinguished names. But resolving the collissions would mean manual processing and this is what we want to avoid.

In the following, the request is augmented with the entry's DN as an operational parameter and routed to the CA that generates the certificate. The certificate is then forwarded to the CMA. This authority puts the certificate on the LDAP into the attribute `hiddenUserCertificate`. It is not published, i.e. written to `userCertificate` at this stage, because the certification process has not yet finished and thus no one should already use the certificate. However, the CMA finishes this step by setting `pkiStatus` to `CERTIFICATION_DONE`.

The entity can now download the certificate from the directory using the client software, store it on the smart card and write it back to the directory into the attribute `userCertificate`. The user is now able to use the PKI's services. In parallel all other participants of the PKI can see and use the certificate. This approach assures that the certificate cannot be used before the enrollment of the entity has finished. It can be enhanced to a proof-of-possession as described in

```
...
validity:
notBefore: 05-05-2005
notAfter: 06-06-2006
subject:
CN=jdoe, OU=student, OU=Computer
Science, O=university, C=DE
...
extensions:
subjectAltName:
rfc822Name: jdoe@cs.university.de
directoryName: CN=jdoe, OU=CS,
OU=student, OU=addresses
CertificatePolicies:
policyIdentifier: idStudentPolicy
```

Figure 3. Example of the to-be-signed certificate for John Doe

[1], if the entity's key-pair can be used for encryption. In this case the certificate would not be put in the attribute `hiddenUserCertificate`, but encrypted with the contained public key and written to `userEncryptedCertificate`. The entity must then be able to decrypt the certificate, i.e. use the correct smart card, in order to have it published.

One thing is still left open for explanation. This is how the public key gets into the certificate request. To be consistent with our scheme, the public key should be put into the directory as well. We therefore introduce the attributes `userPublicKey`, `userPublicKeyAlgorithm` and `userPublicKeyParameters` that may contain the binary encoding of a public key as well as its algorithm ID and parameters. It should either be used, if the entity does not participate in the process in which the key is uploaded or if proof-of-possession is done. Otherwise the authenticity of the public key cannot be guaranteed. To be able to deal with signature-only keys, we have introduced a `userPKCS#10` attribute which replaces `userPublicKey` in this case.

Adding the public key requires to extend the workflow. There are in general two possibilities. Either the public keys are uploaded in a batch process by the trust center or by the entities themselves. In the first case the trust center needs a list that maps the identities of the entities to the public keys on the cards that they have received. This requires that the identity is recorded when the card is sent or handed out to the entity. In any case the authenticity of this mapping is important, since issuing a certificate means to guarantee that the entity owns the private key that corresponds to the respective public key. Having the entity upload the public key requires an additional communication step between the entity and the directory, one for uploading the key and one for fetching the certificate. In this case a proof-of-possession must be conducted. Moreover, the trust center must assure that the public key belongs to one of the cards that were provided to the entities. Otherwise it would not be able to give any guarantees about the quality of the keys and tokens which the keys are stored in. For this reason the RA should obtain a list of valid public keys from the card vendor and match the public keys in certificate requests against this list.

Now all required information is in place and the client application changes `pkiStatus` to `CERTIFICATE_CONFIRMED`. This is done in order to signal to the trust center that the certificate has reached the intended recipient and the process has finished so far. In this process two factors are important for an authentic certificate request. The entity needs to know the passphrase for binding to the directory in order to access either `hiddenUserCertificate` or `userEncryptedCertificate` and to set `pkiStatus` and `userCertificate`. Furthermore, the entity must have the correct smart card, if proof-of-possession is done.

5.2. Requests for Certificate Revocation

The proposed mechanism supports certificate revocation in a very similar way. From the RA's point of view this is done by setting `pkiStatus` to `REVOCATION_REQUESTED`. This can be done by the entities for their own entry. They can bind to their LDAP entry by PKI means or by passphrase and change the value. This step should be supported by the client software. The status can also be changed by an operator of the trust center who was granted access to the entry by means of the directory's access control list.

When the RA recognizes the `pkiStatus` being set to `REVOCATION_REQUESTED`, it posts a revocation request to the CA. This request basically consists of an appropriate `revocationEntry` structure as used in a Certificate Revocation List [9]. The serial number is taken from the certificate of the entry. The revocation date is set to the current time. After posting the request, the RA sets the `pkiStatus` to `REVOCATION_INITIATED` to signal that a revocation is in progress. After the revocation information has been published, the CMA sets `pkiStatus` to `REVOCATION_CONFIRMED`. This signals that the certificate is revoked.

To get from this state to a new certificate will most likely involve manual processing, since the reason for the revocation needs to be considered and resolved.

5.3. Requests for Certificate Renewal

When certificates reach the end of their validity period, they new ones must be issued. The RA can detect this case automatically. The `pkiStatus` must be set to `CERTIFICATE_CONFIRMED`, a user certificate must be present and the current date must be close to `notAfter`. It is important to assure that no revocation request is ongoing for this entry. Otherwise newly issued certificate would overwrite the revocation.

If certificates expire, basically two options can be taken. Either the new certificate is issued for the old public key (recertification), or a new key-pair is generated for which a new certificate is issued (renewal). Which approach to choose, depends on the situation and must be defined by the trust center's policy. They are treated differently by our approach.

In case of a recertification the process differs from the initial certificate creation in that the old certificate can be used as template for issuing the new one and that no proof-of-posession is required. When the RA notices that a recertification is necessary, it sets the pkiStatus of the respective entry to RECERTIFICATION_INITIATED. It then takes the old certificate from the directory, verifies the signature and that the subjectAlternativeName corresponds to the entry. It then takes the public key from the certificate and the other attributes from the directory as described in the initial certification to form a request for the CA. The certificate is produced and put into the attribute hiddenUserCertificate. To signal that the certificate has been produced, the RA sets pkiStatus to RECERTIFICATION_DONE. The entity can now update the smart card, publish the certificate and confirm the process by setting pkiStatus to CERTIFICATE_CONFIRMED. It is notable here that the old certificate will not be replaced, until the entity has updated his smart card. Up to this point the old certificate remains in place and operable.

In case of renewal the process is almost identical to the initial certification. A new smart card is handed out or sent to the entity. The respective public key is written to the directory either by the entity or the trust center and pkiStatus is set to ENTRY_PLAIN.

6. Security Considerations

The entire security of a PKI following our approach is based on the security features of the directory. This concerns both the effectiveness of the access control mechanisms and the processes which lead to the data that is stored in the directory. Controlling the directory means controlling the PKI. We now show, how the attributes of each entry have to be protected.

It is recommended that the processes that establish the data lead to authentic information in the directory. We have assumed that the directory is already in place and that these processes were set up properly. This mirrors the fact that identification and registration of the personell of companies, universities and similar organization is already implemented in a way that satisfies the security needs. Of course these processes have to be reviewed regarding their security properties before applying the proposed mechanism.

First of all, clients who anonymously bind to the directory should only be able to read the "regular" attributes of each entry as needed for the services offered to everyone. This may include the names and addresses, if the electronic addressbook should be publicly available, and the certificates in order to allow sending encrypted messanges. They must not be allowed to modify any entry. The attributes hiddenUserCertificate, encryptedUserCertifcate, pkiStatus, userPublicKey and userPKCS#10 should not be visible to them to prevent that certificates are used before their enrollment has finished.

We now describe the access policy for those entities that authenticate themselves to the directory. We thereby distinguish between access to own entries, i.e. those they did bind to, and to other entries.

An entity must not be able to (unconditionally) manipulate those attributes of the entry that are used to make up the to-be-signed certificate. Otherwise the entity could forge the contents of the certificate. As will be seen in the case study, it is sometimes applicable that entities have influence on the value to which a certain attribute is set. They may e.g. want to choose the e-mail address from a set of possible addresses. In any case, this access must be limited. This can be done by a server side application that temporarily grants access to the respective attributes while monitoring the values which are stored. It is not possible to achieve this with the access control mechanisms provided by LDAP.

The `pkiStatus` attribute must be writable for the entity to be able to launch certification and revocation requests. The same applies to the `userCertificate` as this is demanded by the certification mechanism. In case the public key is uploaded by the entity, write access for `userPublicKey` and `userPKCS#10` must be granted, respectively.

The RA also needs access to the entries of the entities. It needs write access to the attributes `pkiStatus`, `hiddenUserCertificate` and, respectively, `userEncryptedCertificate` to carry out its tasks. Furthermore, it needs to be able to read all attributes that are necessary for creating the certification request for the CA.

Our approach allows for manual administration of all or certain certificates. This is done by granting administrational personell write access to the `pkiStatus` attribute of other's entries. By this means administration can also be distributed or partitioned. The directory is accessible from all over the network. This allows for geographically distributed administration as well. An organization may for example have one employee per department responsible for managing the certificates requests of the coworkers. Also, a network administrator may be responsible for a group of servers. By having proper access to the respective directory entries, the administrator can issue certificate or revocation requests for them.

Special care has to be taken for the revocation mechanism. Since the entity has write access to the `userCertificate` attribute, it is important that the certificate is authentically linked to this entry. We suggest to establish this binding by adding the entry's distinguished name in the directory to the certificate's `subjectAlternativeName` and having the RA check this binding. Otherwise an entity could revoke arbitrary certificates of the PKI by writing the targeted certificate to the entry and setting `pkiStatus` to `REVOCATION_REQUESTED`. The RA would then take this certificate without any checks and produce a revocation request for the CA based on its serial number.

7. Case Study

We have implemented the proposed mechanism in a project at the Technische Universität Darmstadt. This project started in 2004 and was to establish a PKI that allows the students among other things to log on to the university's computers, to have secure access to the Virtual Private Network and protected web sites, to register for exams and get knowledge of the results. Thus the targeted security services were mainly authenticity and non-repudiation. The student's private key should be kept on secure smart cards. The main goal was to keep the management effort at a minimum both during the initial enrolment and the operation for a user group of 20,000 students. The PKI is for now implemented as a prototype. It is planned to be put into operation in October 2005.

We now provide more details about the PKI. In the initial situation the university already operated an X.500 directory. The directory is maintained by two administrators. It already contains all personal and administrative data that we need for issuing certificates. The data is recorded per student during immatriculation. It is regularly updated by the university's administration and can thus be considered to be correct and up-to-date. The students are provided a passphrase that allows them to authentically bind to the directory. They receive this passphrase per mail together with their student ID card. Upon receipt the students must bind to the directory and change the passphrase within a limited period of time. The student can then choose an e-mail address from a set of still available address that are based on combinations of first and last name. These services are provided as SSL/TLS secured web services. Changing the e-mail address is only possible through these web services. The students do not need or have direct write access to the respective LDAP attribute. It is not possible for them to change the e-mail address a second time.

An important fact for modelling the PKI was that the students are unknown to the university until they appear for immatriculation. Thus we are not able to do any preparations that need the knowledge of the student's identity. On the other hand, it is no option to have each student appear a second time for any management issues. Each additional minute counts times 20,000.

We installed a trust center software at the department for network administration. The software consists of three major modules, RA, CA and CMA. The CA is kept offline and exchanges requests and products with the two other components via mobile media. RA and CMA are installed in a protected area and communicate with the X.500 directory over an SSL/TLS secured channel.

We use smart cards for protecting the student's private keys. Each card is equipped with two chips. One is a crypto-processor for the PKI functionality. It contains the student's private key and the root certificate of the PKI as a trust anchor. The second one is a contactless chip for anonymous electronic payment services within the university. The cards are pre-personalized by the manufacturer. It is also responsible for sending the cards to the students per mail. It therefore receives a list that maps the addresses of the students to a pseudonym. The manufacturer in turn provides a mapping from the pseudonyms to the public keys of the cards that have been sent. This mapping is processed in a batch process to put the public keys into the corresponding entries in the directory. The keys on the card are protected by a null-PIN. I.e. the PIN is initially set to zeros and the card is put in a special state, that allows no operation but changing the PIN. The card leaves this state when the PIN is altered and it is impossible to get into this state again. By checking this state upon receipt of the card the student can be sure that no one has used it up to now. This is sufficient for non-repudiation and authentication purposes and spares the additional effort to securely exchange a PIN.

We chose to modify the existing web services to set the `pkiStatus` to `CERTIFICATION_INITIATED` when a student has ascertained the e-mail address. The modification of these services were only minor but significantly improve the queries conducted by the RA. Revocation is supported in two ways. Students are able to revoke their certificate through a web service. Authorization is done by verifying that the students are able to bind to their entry.

We implemented a client application for the students. It allows them to fetch the certificates from the LDAP, store them on the smart card and write them back to the `userCertificate` attribute. The application supports proof-of-possession using the `userEncryptedCertificate` attribute. Since we use RSA keys, we employ this scheme for signature keys as well. The application is capable of checking the null-PIN status of the smart cards. Implementing a dedicated application was less time consuming then integrating the functionality in all PKI-enabled applications that the students are going to use (e.g. e-mail client, web browser, VPN client).

We did benefit from this approach in the following ways:

- The directory was already set up and contained all necessary information. No additional registration step and no manual processing was needed. Thus time and money was saved.
- The processes which deal with the information stored in the LDAP could remain nearly unchanged. No additional personell was necessary and no additional processes had to be established. The whole PKI is maintained by the two operators, that have also been responsible for the directory server.
- The maintainance of the directory was already known to the operators. This knowledge is sufficient for also maintaining the PKI. Therefore, no new skills were required.

Besides the smart card based infrastructure for the students, a second one for the SSL/TLS servers was established. This PKI is already working. It keeps the private keys in software and utilizes PKCS#10 [10] messages to request certificates. It also uses the proposed mechanism with some adaptations.

8. Conclusion and Future Work

We have shown a mechanism to reuse information that is already available in a directory. The mechanism allows to bootstrap a PKI and carry out certificate life-cycle management operations on basis of a directory service. Distributed management and administration of groups is possible. The mechanism can be used with smart card based PKIs. It is modelled as a finite state machine and it is shown that it suffices the requirements for the respective management operations. The applicability of the mechanism is shown in a case study.

The next steps will be to consider variations of this mechanism. It has already proven to be combinable with keys held in software (e.g. conformant to PKCS#12). A future improvement will be to allow multiple certificates per entity. Therefore, each certificate must have its own `pkiStatus` attribute. This can be done by storing each certificate in an entry that is subordinate to the entity's entry in the DIT. A major improvement will be to have the RA also react on changes to the values of an entry's attributes or on whole entries being moved to a different position in the DIT. The vision is here that a directory administrator, if e.g. an employee changes the organizational unit, simply moves the entry to a new branch in the directory. The RA notices this and triggers the required processes to revoke the old and issue a new certificate. This is work in progress. Our approach may further be combined with the Client Update Protocol for LDAP that was proposed by the IETF. Employing this protocol the RA could avoid polling data from the directory. Instead the RA would be noticed upon changes. The protocol is described in [11].

A. Object Class and Attribute Definitions

Here are the definitions of the object class and its attributes that have been introduced in this paper.

```
# object class for pki management
objectclass   ( 1.3.6.1.4.1.8301.3.2.2.1.10.1
    NAME 'pkiManagement'
    SUP top
    AUXILIARY
    MAY (
    hiddenUserCertificate $ pkiStatus $ userPKCS10 $ userPublicKey $
    userPublicKeyAlgorithm $ userPublicKeyParameters))

# blinded certificate
attributetype ( 1.3.6.1.4.1.8301.3.2.2.1.10.2
    NAME 'hiddenUserCertificate'
    EQUALITY certificateExactMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.8
    SINGLE-VALUE )

# status of management operations
attributetype ( 1.3.6.1.4.1.8301.3.2.2.1.10.3
    NAME 'pkiStatus'
    EQUALITY
    integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )

# for uploading a public key with a proof-of-possession
attributetype ( 1.3.6.1.4.1.8301.3.2.2.1.10.4
    NAME 'userPKCS10'
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.5
    SINGLE-VALUE )

# the public key to certify together with required
# parameters and algorithm ID
attributetype ( 1.3.6.1.4.1.8301.3.2.2.1.10.5
    NAME 'userPublicKey'
    EQUALITY octetStringMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
    SINGLE-VALUE )
```

```
attributetype   ( 1.3.6.1.4.1.8301.3.2.2.1.10.6
    NAME 'userPublicKeyAlgorithm'
    EQUALITY objectIdentifierMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
    SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.8301.3.2.2.1.10.7
    NAME 'userPublicKeyParameters'
    EQUALITY octetStringMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
    SINGLE-VALUE )

# encrypted certificate for proof-of-possession scheme taken from [1]
attributetype ( 1.3.6.1.4.1.8301.3.2.2.1.8
    NAME 'userEncryptedCertificate'
    EQUALITY octetStringMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
    SINGLE-VALUE )
```

References

[1] V. Karatsiolis, M. Lippert, and A. Wiesmaier. Using LDAP Directories for Management of PKI Processes. In *Proceedings of Public Key Infrastructure: First European PKI Workshop: Research and Applications, EuroPKI 2004*, volume 3093 of *Lecture Notes in Computer Science*, pages 126–134, June 2004.

[2] RSA. PKCS#12 v1.0: Personal Information Exchange Syntax Standard, June 1999. http://www.rsasecurity.com/rsalabs (24 Jun. 1999).

[3] R. Guida, R. Stahl, T. Bunt, G. Secrest, and J. Moorcones. Deploying and Using Public Key Technology: Lessons Learned in Real Life. *IEEE Security & Privacy*, 2(4):67–71, 2004.

[4] P. Gutmann. Plug-and-Play PKI: A PKI your Mother can Use. In *Proceedings of the 12th USENIX Security Symposium*, pages 45–58, 2003.

[5] C. Adams and S. Farell. Internet X.509 Public Key Infrastructure Certificate Management Protocols. *IETF Request For Comments*, 2510, March 1999.

[6] J. Hodges and R. Morgan. Lightweight Directory Access Protocol (v3): Technical Specification. *IETF Request For Comments*, 3377, September 2002.

[7] J. Myers. Simple Authentication and Security Layer (SASL). *IETF Request For Comments*, 2222, October 1997.

[8] Recommendation X.509 ITU-T. Information Technology - Open Systems Interconnection - The Directory: Authentication Framework. August 1997.

[9] R. Housley, W. Polk, W. Ford, and D. Solo. Internet X.509 Public Key Infrastructure certificate and Certificate Revocation List (CRL) Profile. *IETF Request For Comments*, 3280, April 2002.

[10] RSA. PKCS#10 v1.7: Certification Request Syntax Standard, May 2000. http://www.rsasecurity.com/rsalabs (20 May. 2000).

[11] R. Megginson, M. Smith, O. Natkovich, and J. Parham. Lightweight Directory Access Protocol (LDAP) Client Update Protocol (LCUP). *IETF Request For Comments*, 3928, October 2004.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

On Automatically Detecting Malicious Impostor Emails[1]

Erhan J. Kartaltepe [a], Shouhuai Xu [a]
[a] *Department of Computer Science, University of Texas at San Antonio,*
{ekartalt,shxu}@cs.utsa.edu

Abstract. In this paper we explore the problem we call "malicious impostor emails." Compared with the fairly well-known abuses such as spam and email worms, malicious impostor emails could be much more catastrophic because their payloads may directly target at the victim users' cryptographic keys (via whatever means) and their content—except the malicious payload as an attachment—could look perfectly like a legitimate one. As a first step in dealing with malicious impostor emails, we present a partial solution that mitigates their damage without forcing the involvement of the users.

Keywords. malicious emails, malicious worms, automatic detection, email security, public key infrastructure (PKI) robustness

1. Introduction

Emails have become an indispensable part of most people's daily routines. However, emails were not originally designed as a utility in an adversarial environment, which may explain why there have been so many incidents related to the abuse of emails. Two fairly well-known abuses are spam/phishing emails and email worms. Spam emails are often commercially-motivated messages that are sent to innocent users. Although they have wasted a significant amount of human and machine resources, in general spam emails are benign in the sense that they do not carry harmful payloads. The so-called "phishing" emails could do more harm with the participation of an innocent user who may be fooled into trusting, for instance, a bogus web link and entering his username/password which is thus gleaned by the adversary. Recent email worms could do more damage *automatically* (i.e., without the participation of the victim users), because an infected machine could self-duplicate the email worms to all the users in the address book on the computer, and thus the adversary could launch more advanced attacks such as coordinated Distributed Denial of Service (DDoS).

In this paper we envision and characterize (in Section 2) a class of potentially even more catastrophic attacks implemented via what we call "malicious impostor emails" that would directly target the cryptographic keys stored on the victim machines (via whatever means). As a first step in dealing malicious impostor emails, we present a solution that can mitigate their damage. The solution is called MAUDE, which stands for "Mutiserver Authentic User DEtector" (see Section 3). We discuss the integration of

[1]Supported in part by a grant from the UTSA Center for Infrastructure Assurance and Security.

MAUDE into real-life email systems, and present the preliminary performance analysis result in Section 4. We discuss the related works in Section 5 and conclude the paper in Section 6.

2. The Malicious Impostor Emails Problem

The authors have often received emails that falsely claim to be from some colleagues or friends. These emails were easy (for them) to recognize because the non-attachment content was not something the claimed sender should/would write, and the attachment was quite obviously something they should not execute (or "double click"). This is just a simple example of how easy the email headers (e.g., the sender address) can be faked by even a not-so-sophisticated adversary. This was an alert and inspired us to ask the following question:

> What if the email header and non-attachment content of an email looks perfectly like something the claimed sender would write, while the malicious attachment also looks legitimate so that it will likely be "double-clicked" by the recipient?

Informally, we call such emails "malicious impostor emails." Before we give a formal definition, we need to establish a good understanding of the possibility of the attack and its potential damage.

How are "malicious impostor emails" possible? We believe that several causes make malicious impostor emails possible. First, email headers (e.g., the `From` and `Subject` fields) can be faked easily. Since the `From` field typically plays an important role in a recipient's reaction to an email (e.g., the tendency to trust or distrust it), it would make an attack succeed relatively easily if the adversary faked the `From` field appropriately. What could make this strategy effective from an adversary's perspective is that it is now very easy for an adversary to glean information such as "who would send whom emails" and "who would be in one's email address book" so that the appropriately faked emails might be able to bypass some filters (e.g., those that target spam). The incidents we experienced suggest that email addresses that have appeared on our department's webpage have been abused by the adversary to send an email to Bob while claiming it was sent by Alice. As another example of this type of social-engineering based attack, we observe that it is tremendously easy for an adversary to collect the information about "who have co-authored papers with whom" (e.g., via the DBLP website). This has a severe consequence because it is likely that emails claiming to be from some co-author would pass any countermeasures without much difficulty.

Second, it is possible for an adversary to send a recipient a faked email such that non-attachment content looks perfectly meaningful. This is so because email communications are typically in cleartext, and thus an eavesdropper can have access to legitimate emails. As a toy example, suppose Alice just sent Bob a legitimate email with an email content denoted by α. Then an adversary who has eavesdropped on the communication channel could simply send an email to Bob by faking the header and making the content α followed by something like "PS: attached is a recent picture taken in Hawaii." It is likely that Bob will "double-click" the attachment.

What would be the payload of malicious impostor emails? We are concerned with malicious impostor emails that target at critical information such as cryptographic keys.

In spite of the exciting progress in cryptography in the past years (e.g., threshold cryptography following [6]), it is still crucial to ensure that average users'cryptographic keys be appropriately protected. This is so because, in order for cryptography (or a public key infrastructure) to be widely utilized in real life activities, the large population of users should be assured that their cryptographic keys are nearly as secure as, for instance, their biometrics (such as fingerprints). Otherwise, large-scale deployment of cryptography may become useless, if not doing more harm than good. Towards this end, we believe that malicious impostor emails are an effective means that can be exploited to compromise cryptographic keys (e.g., by embedding a Trojan Horse or Cryptovirus [19]), and that systematic investigation should be conducted so that their damage can be mitigated, if not prevented altogether.

Malicious impostor emails vs. spam/phishing emails: Here we highlight some issues regarding the relationship between malicious impostor emails and spam/phishing emails; we will discuss it in more detail in Section 5. As mentioned before, spam messages are unsolicited emails that typically do not exploit the innocent machines' vulnerabilities, as they are motivated by economical benefit and thus the spammers are still rational. Phishing emails could lead to severe consequences, but only after the innocent users are fooled into clicking embedded web links and entering their usernames and passwords. In contrast, malicious impostor emails are launched by an adversary that does not have to be rational. As a consequence, solutions to spam and phishing emails do not necessarily apply to the problem of the malicious impostor emails. As an extreme example, the idea of a whitelist, which may be effective in mitigating spam, is doomed in dealing with malicious impostor emails, because the adversary can perfectly fake the `From` field so that the email looks perfectly like one from the legitimate sender.

Malicious impostor emails vs. email worms: We also highlight the potential difference between email worms and malicious impostor emails; more details are discussed in Section 5. Email worms often launch destructive attacks such as Distributed Denial of Service (DDoS), and typically try to infect as many machines as soon as possible. On the other hand, malicious impostor emails may only be interested in breaking certain victim machines pre-selected by the adversary. To this end, malicious impostor emails may corrupt some users' computers as their "step stones" and may not actually do any harm until they have reached the targeted victims, and could adopt even more advanced and crafty strategies to bypass or compromise the deployed countermeasures (e.g., they may spread slowly in a steganographic way).

Summary: Suppose an email consists of three parts: a *header* (including the `From` field), a *non-attachment content* (i.e., the email body that is not part of the attachment), and an *attachment*. For a given email, denoted by EMAIL, let $\mathsf{sender}(\text{EMAIL})$ be the sender id (i.e., the value in the `From` field of the header). For each email user U, let $\mathsf{whitelist}_U$ be the whitelist of email addresses that will be accepted by U, Filter_U be a filter that takes as input an email, analyzes its non-attachment content, and outputs a decision on whether it thinks the email is suspicious (e.g., a spam or phishing email), $\mathsf{wormScanner}_U$ be a worm/virus scanner that takes as input an email, analyzes its attachment, and outputs a decision on whether the attachment is suspicious. We assume that Filter_U is perfect, meaning that any spam or phishing email will be detected. However, $\mathsf{wormScanner}_U$ is not perfect, meaning that it can only detect the malicious attachments that possess known signatures; they have limited success in dealing with polymorphism worms/virus and cannot deal with unknown (zero-day) worms.

Definition 1 (malicious impostor email) *A malicious impostor email is an email, denoted by* EMAIL*, sent to a recipient* U *with* $(\mathsf{whitelist}_U, \mathsf{Filter}_U, \mathsf{wormScanner}_U)$ *such that*

1. $\Pr[\mathsf{sender}(\text{EMAIL}) \in \mathsf{whitelist}_U] = 1$*, meaning that the email can always perfectly fake the email header information including the sender's email address and perhaps the IP addresses incorporated in the email header.*
2. $\Pr[\mathsf{Filter}_U(\text{EMAIL})$ *outputs "suspicious"*$] = 0$*, meaning that the non-attachment content is perfect and cannot even be detected by a human being.*
3. $\Pr[\mathsf{wormScanner}_U(\text{EMAIL})$ *outputs "suspicious"*$] = \theta$ *for some* $0 \le \theta < 1$.

The ultimate goal is to allow automatic detection of *all* malicious impostor emails so that they can be appropriately processed, although in what follows we are only able to present a partial solution called MAUDE.

3. MAUDE: Multiserver Authentic User DEtection

3.1. Model

We consider a system consisting of multiple email servers. Each server has a set of legitimate users that can utilize its service. We call the *outgoing* email server the *physical* machine which actually initiated the sending of a sender's email, and the *incoming* email server the *physical* machine which delivers a message to the receipient. Typically, an email takes a path consisting of at least one email server. That is, a path starts with the sender's physical email server and ends at the recipient's physical email server; the servers on the path between may be called relays). Then, the *first* server on the path is the outgoing server, and the *last* server on the path is the incoming server. Therefore, we are only interested in the outgoing and incoming servers, and will ignore the relays.

We assume that (a subset of) the servers have some established trust, which may be fulfilled by each pair of email servers sharing a common secret (i.e., a cryptographic key). This would be feasible if the system of interest is under the same administrative domain (e.g., a campus network or the network of a state university system); We will discuss more on this assumption below.

We consider a probabilistic polynomial-time adversary that intends to send *malicious impostor emails* with the intent that the recipients will "double-click" the attachment.

The basic idea underlying MAUDE is very simple. When a recipient's email server (i.e., the incoming email server) receives an email with an attachment that claims to be from some server (i.e., the alleged outgoing email server), the incoming email server will contact the alleged outgoing email server to verify that it sent that specific email. If so, the email is processed as in the original email system; otherwise, the email is suspicious and appropriate action is taken (e.g., thrown into some special folder with an explicit alert or even deleted).

3.2. Detailed Description

At an abstracted level, MAUDE consists of several (distributed) algorithms. Before we present the algorithms, we need to introduce some notations. Let $sender$ be the sender of an email and $recipient$ be the recipient of an email, both in the form of the string "$user@domain$", where $user$ is the username and $domain$ is the hostname concatenated with "." and the top-level domain name. Let $subject$ be the subject of the email and $body$ be the email content itself, containing both the text message (if present) and the attachment.

The above values can be obtained from the real-life email system, namely Sendmail in our case study, via the following algorithm getMaudeMessage. This algorithm's design is based on the observation that Sendmail prepares the SMTP payload as a list of headers followed by the email message. A header has two members $field$ and $value$, where $field$ holds the type of header and $value$ holds its content. Let $Headers$ be the set of headers in the email and $email$ be the email message itself. The following algorithm takes as input $(Headers, email)$, which is provided by Sendmail, and returns the desired $(sender, recipient, subject, body)$.

```
getMaudeMessage($(Headers, email)$)
    FOREACH $header \in Headers$
        IF $(header.field =$"From"$)$
            $sender \leftarrow header.value$
        IF $(header.field =$"To"$)$
            $recipient \leftarrow header.value$
        IF $(header.field =$"Subject"$)$
            $subject \leftarrow header.value$
    $body \leftarrow email$
    Return $(sender, recipient, subject, body)$
```

Let each outgoing email server maintain a database DB for the emails it has sent. An entry of DB is of the format $(sender, recipient, \mathsf{H}(sender, recipient, subject, body))$. Each outgoing or incoming email server maintains a key tables of entry format $(peer, key)$, where $peer$ is another server which shares its common secret key. Each server also runs a MAUDE process which will operate over the DB.

In order fulfill its purpose, MAUDE will treat the following building-block algorithms as black-box in nature, but whose functionalities are well-understood:

- host$(emailAdd)$ returns the hostname of the email address.
- domain$(emailAdd)$ returns the domain name of the email address.
- getKey$(machine)$ returns the key shared between the server that is running this algorithm and the remote sender/recipient server called $machine$.
- oldSend$(sender, recipient, subject, body)$ is the original email sending functionality (e.g., the one implemented by Sendmail).
- oldReceive$(sender, recipient, subject, body)$ is the original email receiving functionality (e.g., the one implemented by Sendmail).
- impostor$(sender, recipient, subject, body)$ handles the impostor emails according to a policy (e.g., placing them in a special folder or deleting them).
- H is a collision-resistant hash function.
- HMAC is a secure message authentication code such as HMAC-SHA1 [4].

MAUDE consists of the following algorithms.

- addMaude$(sender, recipient, \alpha)$ updates DB with a new entry. This algorithm is initiated by the outgoing server and executed by its MAUDE.
- queryMaude1$(sender, recipient, subject, body)$ determines whether the entry $(sender, recipient, subject, body)$ appears in DB. This algorithm is initiated by the incoming server and executed by its MAUDE.
- queryMaude2$(sender, recipient, subject, body)$ asks the local MAUDE to check if the email was sent by the claimed server. This algorithm is initiated by the incoming server and executed by its MAUDE.
- queryPeerMaude$(sender, recipient, subject, body, \alpha, r, \beta)$ is an interactive algorithm between an incoming MAUDE and an outgoing MAUDE. It is initiated by the incoming server's MAUDE and executed by the remote outgoing server's MAUDE.
- newSend$(sender, \{recipient_i\}_{1 \leq i \leq n}, subject, body)$ is the extended email-sending algorithm run by the outgoing server.
- newReceive$(sender, recipient, subject, body)$ is the extended email-receiving algorithm run by the incoming email server.

Their precise functionalities are described below.

```
addMaude$(sender, recipient, \alpha)$
    $\mathsf{DB} \leftarrow \mathsf{DB} \cup \{(sender, recipient, \alpha)\}$

queryMaude1$(sender, recipient, \alpha)$
    IF $((sender, recipient, \alpha) \in \mathsf{DB})$ THEN
        $\mathsf{DB} \leftarrow \mathsf{DB} \setminus \{(sender, recipient, \alpha)\}$
        Return TRUE
    ELSE  Return FALSE

queryMaude2$(sender, recipient, \alpha)$
    $k \leftarrow \mathsf{getKey}(\mathsf{host}(sender))$
    select a random $r$
    $\beta \leftarrow \mathsf{HMAC}(k; sender, recipient, \alpha, r, 0)$
    $\gamma \leftarrow \mathsf{queryPeerMaude}(sender, recipient, \alpha, r, \beta)$
    IF $(\gamma = \mathsf{HMAC}(k; sender, recipient, \alpha, r, 1))$ THEN
        Return TRUE
    ELSE Return FALSE

queryPeerMaude$(sender, recipient, \alpha, r, \beta)$
    IF $((sender, recipient, \alpha) \notin \mathsf{DB})$ OR $(\mathsf{getKey}(\mathsf{host}(recipient)) = \perp)$THEN
        Return $\perp$
    ELSE
        $k \leftarrow \mathsf{getKey}(\mathsf{host}(recipient))$
        IF $(\mathsf{HMAC}(k; sender, recipient, \alpha, r, 0) = \beta)$ THEN
            $\mathsf{DB} \leftarrow \mathsf{DB} \setminus \{(sender, recipient, \alpha)\}$
            Return $\mathsf{HMAC}(k; sender, recipient, \alpha, r, 1)$
        Return $\perp$
```

```
newSend$(sender, \{recipient_i\}_{1 \le i \le n}, subject, body)$
    FOR $i = 1$ to $n$
        IF (getKey(host$(recipient_i)$) $\neq \perp$) THEN
            addMaude$(sender, recipient_i, \mathsf{H}(sender, recipient_i, subject, body))$
    oldSend$(sender, \{recipient_i\}_{1 \le i \le n}, subject, body)$

newReceive$(sender, recipient, subject, body)$
    IF (host$(sender)$ = host$(recipient)$) THEN
        IF (queryMaude1$(sender, recipient, \mathsf{H}(sender, recipient, subject, body))$)
            THEN oldReceive$(sender, recipient, subject, body)$
            ELSE impostor$(sender, recipient, subject, body)$
    ELSE  IF (getKey(host$(sender)$) $\neq \perp$) THEN
              IF (queryMaude2$(sender, recipient, \mathsf{H}(sender, recipient, subject, body))$)
                  THEN oldReceive$(sender, recipient, subject, body)$
                  ELSE impostor$(sender, recipient, subject, body)$
    ELSE oldReceive$(sender, recipient, subject, body)$
```

3.3. Analysis

Security of MAUDE against a probabilistic polynomial-time adversary is clear (based on the security of the message authentication codes).

The extra computational overhead imposed on the servers is small (i.e., computing some message authentication tags). The size of the database DB grows proportionally to the number of email-sending requests, and decreases as the emails have been confirmed. Thus, on average the size of DB is relative to the difference between the email-sending requests and the email-confirmation requests.

3.4. Discussion on Assumptions and Design Rationale

We observe that MAUDE's utility applies when there is some well-established trust between the email servers. This is reasonable for email servers that are under the same administrative jurisdiction, or emails servers that are administered by some collaborative partners (e.g., all the schools of a state university system). This already suffices to solve the malicious impostor email problem in the setting that the "claimed" senders (i.e., the senders listed in the headers of the malicious impostor emails) are indeed affiliated with the email servers that are administered by the collaborative partners.

What if the servers do not have the above-mentioned initial trust relationship? An immediate approach is to assume the existence of a (potentially small-scale) PKI that certifies the public keys of the email servers in one way or another. Based on this, there are numerous cryptographic protocols that suffice to establish initial trust and common keys between the servers. If it is possible to establish such a "meta" PKI, we can utilize it to achieve a robust large-scale PKI so that the large population of average users' (rather than the small-population of servers') keys can be well-protected from malicious impostor emails.

But if we assume the existence of the initial trust, why don't we simply let each outgoing server associate with every email a digital signature or message authentication tag? The reasons against this are threefold. First, we believe that an ultimate solution should not be solely based on the existence of such an established, though small-scale, trust structure. Under certain circumstances, it would still be useful even if we can achieve *best effort* security in the following sense: An incoming server still contacts an outgoing email server to check if a claimed email was indeed sent by it, even if this communication is *not* protected by a cryptographic means (due to the absence of a common key). In this case, the communication initiated by the incoming email server could become another factor of authentication (or the adversary has to use some real email server with real domain name and IP address, which would help forensics investigation, or the adversary has to hijack many TCP sessions, which may significantly limits its effectiveness).

Second, we are investigating methods that can help a pair of servers that can establish a certain degree of trust without relying on any trusted third parties. Of course, the established trust is weaker than the one established via some trusted third parties; however, this is still promising because the trust may now be exclusive between the two servers.

Third, the way the real-life email systems operate is quite complicated. For example, it is rather difficult for an outgoing email server to figure out which *physical* machine is the real destination machine; on the other hand, it is relatively easy for the real destination machine to determine which *physical* machine was the sender (initiator) of the email. Therefore, the design makes it possible for the destination server to *only* contact the initial sending server; this avoids involving the numerous email relays that are indicated in the email headers (which could be quite long), and makes the scheme practical.

4. Integrating MAUDE into Email Systems: Experiments, Performance, and Effectiveness

4.1. Methodology

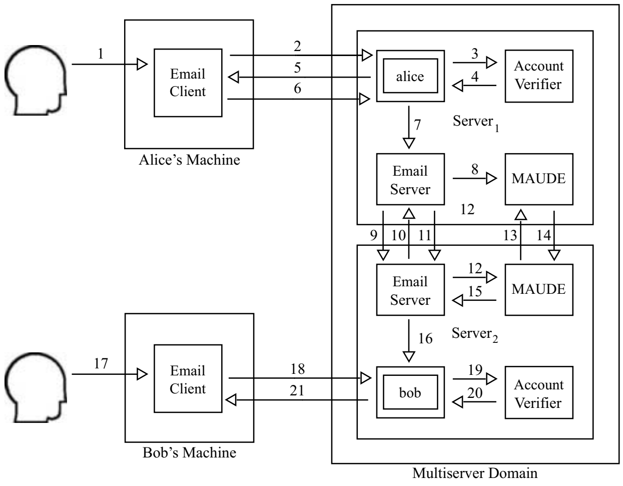


Figure 1. A low-level view of MAUDE

In order to test the effectiveness of MAUDE, we implemented it into Sendmail, a widely deployed email system. To avoid a modification of the SMTP protocol or delay the communication between the two email servers, the modified Sendmail server waits until the SMTP protocol has completed before extracting the headers and constructing the message it sends to its local MAUDE. In other words, we devise a scheme to modify Sendmail while encapsulating the SMTP protocol from MAUDE. As a result, an email system with MAUDE operates as follows (see also Figure 1).

1. Alice logs into her local email client.
2. After composing a message, Alice's email client logs in to the server.
3. Alice's password is sent to an account verifier to determine her authenticity.
4. The account verifier grants Alice access to her account.
5. The server contacts the email client to send the message.
6. Alice's email client sends the message to the outbox in her account.
7. The system delivers the message to the email server for delivery.
8. Alice's email server executes getMaudeMessage$(Headers, email)$ which returns $(sender, recipient, subject, body)$. Immediately after, Alice's email server executes addMaude$(sender, recipient, \mathsf{H}(sender, recipient, subject, body))$.
9. Alice's email server server contacts Bob's email server to transmit the message.
10. Bob's email server accepts the message, recording the other server's hostname and IP address.
11. Alice's email server delivers the message.
12. Bob's email server executes getMaudeMessage$(Headers, email)$ which returns $(sender, recipient, subject, body)$. Immediately after, Bob's email server executes queryMaude2$(sender, repicient, \mathsf{H}(sender, recipient, subject, body))$.
13. Bob's MAUDE activates queryPeerMaude$(sender, recipient, \alpha, r, \beta)$.
14. Alice's MAUDE executes queryPeerMaude$(sender, recipient, \alpha, r, \beta)$ and returns γ.
15. Bob's MAUDE returns TRUE (supposing queryPeerMaude returns the correct value).
16. The email server routes the message to Bob's inbox.
17. Bob logs into his local email client.
18. To retrieve his email, Bob's email client logs in to the server.
19. Bob's password is sent to an account verifier to determine his authenticity.
20. The account verifier grants Bob access to his account.
21. The server delivers the email in Bob's inbox to the email client.

4.2. Metrics

The metrics we are interested in are the following:

- The delay incurred by MAUDE on the outgoing email server. The delay is defined as the time period between the point that Sendmail has accepted the message for delivery and is about to contact its local MAUDE and the point that the local MAUDE has updated the database.
- The delay incurred by MAUDE on the incoming email server. The delay is defined to be the time period between the point that the SMTP protocol has begun and the point that the email is thrown into the email box (or discarded).
- The effectiveness and accuracy of MAUDE's decisions.

4.3. Experimental System Settings

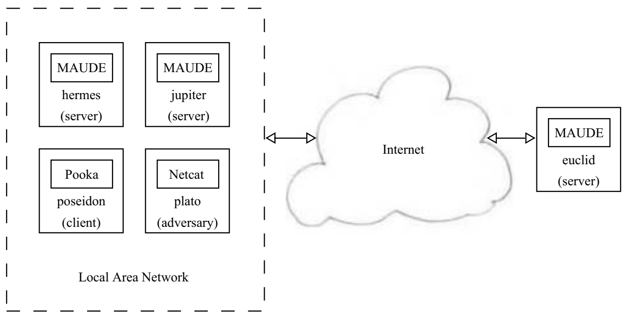


Figure 2. Integrating MAUDE into real-life email systems

The system environment is depicted in Figure 2. The email servers are within a university campus network, and the email clients are both within and outside the campus network. The email servers are called `hermes` and `jupiter`, respectively. There are two email client machines: `poseidon` acted as a friendly external computer within the LAN with authorized access to `hermes` through an email client, and `plato` was an adversary client machine within the campus network. The purpose of `plato` is to send faked emails but make them look as though they were sent by legitimate users through a tool like Netcat. A fifth machine, `euclid`, tested the performance of MAUDE on a non-dedicated internet connection. The three servers, `hermes`, `jupiter`, and `euclid` recognized each other by sharing some pair-wise keys. Figure 3 reviews the concrete configurations of the machines and networks.

Machine	Processor	Internet Connection	Relavant Software
`hermes`	2.26 GHz P4	Gigabit LAN	Sendmail 8.12.9, MAUDE
`jupiter`	2.80 GHz P4	Gigabit LAN	Sendmail 8.12.9, MAUDE
`poseidon`	2.26 GHz P4	Gigabit LAN	Pooka Email Client
`plato`	2.80 GHz P4	Gigabit LAN	Netcat, mutt
`euclid`	2.40 GHz P4	100 Megabit Cable	Sendmail 8.12.9, MAUDE

Figure 3. System Settings

MAUDE was written in Java 1.5.0 and ran on the Java Virtual Machine. For processing incoming and outgoing emails, Java's crypto library was used to compute any hash value or HMAC it needed, in both cases using the SHA1 algorithm. For testing, a program simulating Sendmail on each server sent valid messages to its local MAUDE. This program, implemented in C, used Peter Gutmann's cryptlib 3.1 library to compute the SHA1 hash value for the tuples it sent to MAUDE. This simulator measured the extra delay time MAUDE added to the delivery of an email.

4.4. Performance

To examine the delay incurred by MAUDE on the sender's email server, time was marked before and after each transmission over 10000 requests. We repeated this test ten times and took the average over the runs. Figure 4 shows the trends over the 10000 requests.

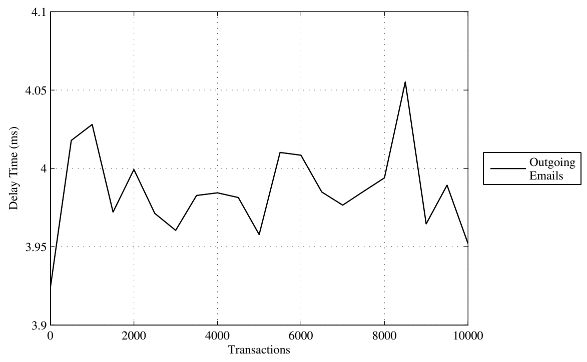


Figure 4. Outgoing Emails over 10000 trials

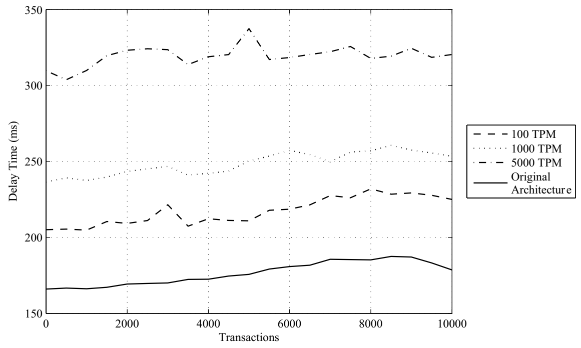
For incoming emails, we ran three tests of 10000 concurrent requests at the rates of 100, 1000, and 5000 requests per minute, respectively. These stress tests were implemented to show how MAUDE operates with a high-volume incoming email server. For each of the three experiments, we obtain the corresponding delay time by averaging 10 independent runs (Note that a single run of the simulation at the rate of 100 requests per minute requires 100 minutes for all 10000 requests). In plotting the data, we fit the curves by taking a moving average of 500 requests for a clear representation of each trend. We plotted the delay time for the three stress tests for the LAN and WAN MAUDE in Figure 5.

In the LAN experiments, the Sendmail simulator on `jupiter` delivered a valid message to the local MAUDE. In turn, the local MAUDE contacted `hermes` with a valid message to determine if `hermes` sent the message. We recorded the time it took for `hermes` to respond to each request from `jupiter` from the moment Sendmail constructed the message for MAUDE until after it processed MAUDE's return message. In the WAN experiments, the simulator on `euclid` operated in the same fashion.

Delay times taken as an average are summarized in Figure 6. The time for Sendmail to send an email averaged under 177 milliseconds. At the rate of 100, 1000, and 5000 requests per minute, the delay time increase due to a LAN MAUDE is increases by 23.8%, 40.6%, and 81.1%, respectively. However, even via a WAN, MAUDE's delay time is less than the original architecture's processing capability.

4.5. Effectiveness

Using the Pooka email client [15] on `poseidon`, the email client showed no noticeable delay when MAUDE was activated versus when it was absent. Since MAUDE's effects



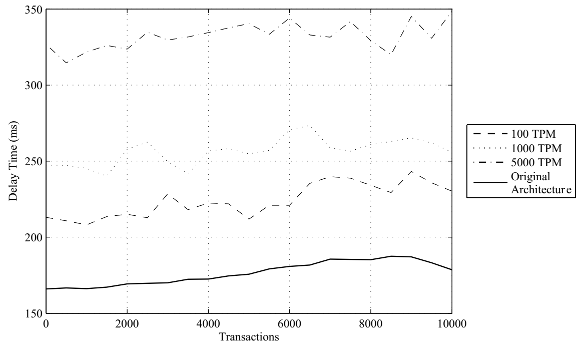


Figure 5. Incoming emails over 10000 trials via a LAN MAUDE (top) and WAN MAUDE (bottom)

Transactions Per Minute	Original Architecture	Incoming Email (LAN Network)	Incoming Email (WAN Network)
100	176.406	217.307	224.061
1000	176.406	248.619	256.440
5000	176.406	319.491	332.351

Figure 6. Average delay times for each experiment (milliseconds)

are encapsulated away from any email agent implementation, no special setup to the client was necessary.

When using Netcat [14] on `plato` to send bogus email messages to `hermes`, every email was checked against the supposed sender's MAUDE, and each came back as fraudulent. This was the case when using the more user-friendly mutt email client [13] to send impostor emails as well. Moreover, every email sent by a legitimate user was not flagged as a possible impostor. Thus, we note that no false positive or false negative is possible.

5. Related Work

On the relationship between malicious impostor emails and PKI. In order for a public key infrastructure (PKI) to achieve its fully-expected utility, it is necessary to ensure that the cryptographic keys are appropriately protected. Advanced cryptographic mechanisms such as threshold cryptography [6] have been invented to protect critical cryptographic functionalities (e.g., the signing function of a certificate authority) so that the damage incurred by sophisticated attackers could be mitigated. We observe, however, that the protection of average users' cryptographic keys would be an equally important factor in deploying a PKI. Unlike in the setting of the servers, the protection of individual cryptographic keys could be far more challenging (given that smartcards are not widely deployed), particularly because the individual users' machines are always used to fulfill their routing tasks without being administered by professionals (servers are relatively well-protected by professionals who may keep patching the relevant software programs). Moreover, there is a wide spectrum of ways to compromise an individual's machine (and thus the cryptographic keys). One such method is the above described "malicious impostor emails" whose malicious payloads can compromise one's private keys (e.g., [19]). This explains why dealing with malicious impostor emails is an essential part of protecting a PKI. On the other hand, it would be ideal if MAUDE can get support from a small-scale PKI (e.g., one for the email servers of a relatively small population), then a large-scale PKI (involving a large population of average users) built on it can be better protected—this is the case of MAUDE.

More on malicious impostor email vs. email worms. Current worms (e.g., [16,21,18, 11,17,22]) are typically high-speed because they want to infect as many machines *as fast as possible*. Since "high-speed spreading" does not necessarily apply to malicious impostor emails, they need new solutions.

More on malicious impostor emails vs. spam/phishing. Spam emails are quite different from malicious impostor ones, and therefore solutions to countering the former might not be effective at combating the latter. As a consequence, an approach founded on economic incentives that could be effective against spam would not necessarily translate to resolving the problem of malicious imposter emails. Goodman [10] analyzed the problem of preventing outgoing spam while assuming that the adversary is *rational*, but this solution would not work against malicious emails because the adversary launching them does not have to be rational.

Machine learning or data mining based filters [9] for detecting or blocking potential spam are not necessarily effective in detecting or blocking malicious impostor emails on their own. This is so because the email content, besides the offending attachment, could be an otherwise perfectly legitimate email.

The puzzle approach, based on computational puzzles [8] or their recent variants called "moderately hard, memory-bound functions" [2,7], could be effective in dealing with spam. Similarly, the related virtual stamps or other payment schemes (e.g., [1,12]) could be successful as well against spam, provided that spammers have only a reasonable budget (recall that spammers are typically motivated by economic incentives). However, they do not solve the problem of malicious impostor emails as they involve the end (or average) users. Moreover, introducing cryptography-based stamps indeed bring in new problems that the end users need to protect their cryptographic keys appropriately. We need a solution that is transparent to end users.

On the relationship with other loosely related works. There are some other works that are loosely related to ours. For example, certified emails [3,5,20] deal with *fair exchange* between a recipient's receipt and a sender's email. Although there may be some resemblance between the certified email protocols and our approach to dealing with impostor emails, the two problems are indeed fundamentally different. First, the ultimate goal of certified emails is to fulfill fair exchange transactions between two *end* users, not necessarily between the corresponding incoming and outgoing email servers (unless one would like to make the unrealistic assumption that the users fully trust, and thus would even give their private keys to, the email servers). In contrast, our approach to dealing with impostor emails is contained to the interactions between the email servers. Indeed, such an ideal solution, as what we have proposed, should be transparent to the end users. Second, in contrast to the case of certified emails, the sender of malicious impostor emails (i.e., the adversary) never has the intent to conduct a fair exchange with a (victim) recipient. Indeed, the adversary always tries to hide itself so that it will not be held accountable.

6. Conclusion and Future Work

We introduced the problem we call "malicious impostor emails"—emails that can perfectly fake email headers and possess non-attachment content that can flawlessly mimic legitimate ones. We explored a partial solution to this problem as a first step toward countering this powerful attack. We are continuing the investigation in the following directions:

- How can we remove or weaken the reliance on the initial trust between the email servers? This is particularly relevant when we consider the Internet as a single system.
- Our solution is effective provided that every involved email server has MAUDE installed. The hope is that MAUDE can be employed incrementally. So can we establish an analytical model to understand the effectiveness of the incremental deployment of MAUDE?

Acknowledgements

We are grateful to Allen Petersen for the use of the Pooka email client source code and insight into the program. We also thank Carlos Cardenas for his suggestions, and Allen Fouty and Hugh Maynard for their helpful comments and feedback. Finally, we thank the anonymous reviewers for their useful comments.

References

[1] M. Abadi, A. Birrell, M. Burrows, F. Dabek, and T. Wobber. Bankable postage for network services. In *Advances in Computing Science—ASIAN 2003, Programming Languages and Distributed Computation, 8th Asian Computing Science Conference*, pages 72–90.

[2] M. Abadi, M. Burrows, M. Manasse, and E. Wobber. Moderately hard, memory-bound functions. In *Proceedings of the 10th Annual Network and Distributed System Security Symposium*, pages 25–39.

[3] A. Bahreman and J. D. Tygar. Certified electronic mail. In *Proceedings of the 1994 Network and Distributed Systems Security Conference*, February 1994, pages 3–19.

[4] M. Bellare, R. Canetti, and H. Krawczyk. Keying hash functions for message authentication. In *Proceedings of Crypto'96*.

[5] R. Deng, L. Gong, A. Lazar and W. Wang. Practical Protocols For Certified Electronic Mail. *Journal of Network and System Management*, vol. 4, no. 3, 1996.

[6] Y. Desmedt and Y. Frankel. Threshold Cryptosystems.*CRYPTO'89*.

[7] C. Dwork, A. Goldberg, and M. Naor. On memory-bound functions for fighting spam. In D. Boneh, editor, *Proc. CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 426–444. Springer-Verlag, 2002.

[8] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In Ernest F. Brickell, editor, *Proc. CRYPTO 92*, pages 139–147. Springer-Verlag, 1992. Lecture Notes in Computer Science No. 740.

[9] J. Goodman and R. Rounthwaite. Smartproof. manuscript, 2004.

[10] J. Goodman and R. Rounthwaite. Stopping outgoing spam. In *Proceedings of ACM E-Commerce 2004*, 2004.

[11] A. Gupta and R. Sekar. An Approach for Detecting Self-Propagating Email Using Anonmly Detection. *Proceedings of RAID'03*.

[12] A. Herzberg. Controlling spam by secure internet content selection. In *Proceedings of the 4th Conference on Security in Communication Networks*, *Lecture Notes in Computer Science*, 2004.

[13] Michael Elkins The Mutt E-Mail Client `http://www.mutt.org/`

[14] Giovanni Giacobbi The GNU Netcat Project `http://netcat.sourceforge.net`

[15] A. Petersen Pooka: A Java Email Client. `http://suberic.net/pooka`

[16] S. Staniford, V. Paxson, and N. Weaver. How to Own the Internet in Your Spare Time. *Proceedings of Usenix Security 2002*.

[17] C. Wong, S. Bielski, J. McCune, and C. Wang. A Study of Mass-Mailing Worms. *Proceedings of ACM Worm 2004*.

[18] J. Wu, S. Vangala, L. Gao, and K. Kwiat. An Effective Architecture and Algorithm for Detecting Worms with Various Scan Techniques. *Proceedings of NDSS 2004*.

[19] A. Young and M. Yung. Cryptovirology: Extortion Based Security Threats and Countermeasures. *IEEE Symposium on Security and Privacy'96*.

[20] J. Zhou and D. Gollmann. Certified electronic mail. In *Proceedings of the Fourth European Symposium on Research in Computer Security (ESORICS 96)*, pages 160–171.

[21] C. Zou, L. Gao, W. Gong, and D. Towsley. Monitoring and Early Warning for Internet Worms. *Proceedings of ACM CCS'03*.

[22] C. Zou, D. Towsley, and W. Gong. Email Worm Modeling and Defense. *Proceedings of International Conference on Computer Communications and Networks (ICCCN'04)*.

Non-Repudiation

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Generic Fair Non-Repudiation Protocols with Transparent Off-Line TTP

Guilin Wang [1]
Institute for Infocomm Research (I^2R)

Abstract. A non-repudiation protocol is aimed for exchanging a digital message and an irrefutable receipt between two mistrusting parties over the Internet. Such a protocol is said *fair*, if at the end of any possible protocol execution, either both parties obtain their expected items or neither party does. In this paper, we first argue that it is really meaningful in practice to exploit *generic* fair non-repudiation protocols with *transparent* off-line TTP. Namely, in those protocols, each involved party could use *any* secure digital signature algorithm to produce non-repudiation evidences; and the issued evidences are the same regardless of whether the TTP is involved or not. Then, we present such a fair non-repudiation protocol to overcome the limitations and shortcomings in previous schemes. Technical discussions are provided to show that our protocol is both secure and very efficient. In addition, some extensions are also pointed out.

Keywords. non-repudiation, certified e-mail, fair exchange, security protocol

1. Introduction

1.1. Background

Non-repudiation service is essential for many electronic transactions, where irrefutable evidences need to be generated, exchanged, and validated via computer networks. After the completion of such a transaction, each involved party should obtain the expected items. Therefore, if any dishonest party denies his/her participation in a specific transaction, others can refute such a claim by providing electronic evidences to a judge.

A non-repudiation protocol allows two potentially mistrusting parities to exchange a digital message and the evidence of origin for an undeniable receipt over the Internet in a *fair* way, i.e., each party gets the other's item, or neither party does. We assume that the sender Alice wants to deliver a digital message M to the receiver Bob but with the guarantee that Bob can access the content of M if and only if she obtains a receipt from Bob showing that Bob has already received M. At the same time, Bob is also convinced that he is able to get the message M and the evidence of origin from Alice when he issues such an undeniable receipt to Alice. While non-repudiation evidences could be provided by some standard cryptographic primitives such as digital signatures, fairness is much more difficult to be achieved [16,20].

[1]Correspondence to: Guilin Wang, Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613. Tel.: +65 6874 1954; Fax: +65 6775 5014 ; E-mail: glwang@i2r.a-star.edu.sg.

Technically speaking, non-repudiation protocol belongs to a wider topic: fair exchange. Namely, how two (or multiple) mutually mistrusted parties exchange digital items over computer networks in a fair way. Actually, fair exchange includes the following different but related issues: non-repudiation protocols [26,27,21,20,16], certified e-mail systems [4,14,23], fair exchange of digital signatures [11,5,3], contract signing schemes [1,23,6], and e-payment solution [8]. In a certified e-mail scheme, a sender Alice wants to deliver a digital message to a receiver Bob with the guarantee that Bob can access the content of the e-mail *if and only if* Alice obtains an irrefutable receipt from Bob. So, we know that the purposes of non-repudiation protocols and certified e-mail schemes are almost the same, except that a certified e-mail scheme may not provide evidence of origin. For more references and discussions on the relationships among those variants of fair exchange, please refer to [2,16]. In this paper, we shall focus on the topic of non-repudiation protocols and certified e-mail schemes.

1.2. Related Work

Since direct fair-exchange between two parties is extremely inefficient on both aspects of computation and communication [11], realistic implementation of non-repudiation protocol needs a trusted third party (TTP), though the extent of the TPP's involvement may be different. A simple but unsatisfactory solution [9] is to exploit the TTP as a *delivery authority* , i.e., two involved parties generate and submit proper digital items to the TTP, then the TTP checks the correctness of those items and forwards each item to the corresponding recipient. The shortcoming is that the TTP is likely to become a bottleneck in the system, due to the fact that it is involved in every step of exchange. An ingenious idea is proposed in the protocol of Zhou and Gollmann [26] by using the TTP as a *light-weight notary*. That is, in their scheme, only a short symmetric key is forwarded and notarized by the TTP, while the whole message in ciphertext is directly transferred to the receiver. However, both of the above two schemes are actually inefficient and expensive in the real world because an *on-line* TTP is required, i.e., the TTP takes part in every execution (though may not in every step). The point is that the TTP needs to be paid for all provided services.

A more appealing and practical approach is to design non-repudiation protocols with an *off-line* TTP [27]. Actually, such schemes are also called *optimistic* [1,2], since the TTP is not invoked in the execution of exchange unless one of the two parties misbehaves or the communication channel is out of order. Along this promising and interesting approach, a number of protocols have been proposed in the literature [28,29,21,16]. Later, Gürgens et al. [19,20] intensively studied the security of those schemes. According to their results, the desirable fairness may not be guaranteed under some subtle but reasonable attacks: (1) The protocol labels in [26,27] need to be changed in a small but important way; (2) The schemes in [21,16] allow a dishonest receiver to access the message without issuing a receipt to the sender (so unfair for the sender)[1]; and (3) There is a potential weakness in the solution proposed in [28,29]. In other words, all those schemes have security weaknesses in different senses. In [20], a new non-repudiation protocol is further proposed to avoid those security shortcomings.

[1]More specifically, Gürgens et al. just identified such an attack against the scheme [16]; but we note that this attack works in the scenario of [21], too.

In [23], Micali proposed simple certified e-mail schemes with *transparent* TTP (*in-viable post office*, in his terminology). This means that the generated non-repudiation evidences are the same regardless of whether the TTP is involved or not. However, we remark that Micali's schemes have three weaknesses. First, there are no evidences for non-repudiation of origin (NRO), but in some scenarios the receiver may need NRO to prove the message origin. Second, his schemes are inefficient if the delivered message is long, since in his schemes messages are encrypted under asymmetric instead of symmetric encryption algorithms. About this shortcoming, Micali did pointed out that messages could be encrypted alternatively with a symmetric key K, while K is encrypted again under the receiver's public key. Actually, this idea is adopted by most of existing non-repudiation protocols [26,27,28,29,21,16,14,20]. The point is that this conversion is *not* trivial, according to the relevant researches [19,20]. Finally, there is a potential attack, where the sender Alice could cheat the receiver Bob by mixing identities of different TTPs. Consequently, Alice will get a valid receipt, but Bob perhaps cannot receive the message. Such an attack is meaningful in practice since it seems unreasonable to assume that Bob knows the existence of all TTPs and may contact each TTP individually for help.

Another two interesting certified e-mail schemes with transparent TTP are proposed in [14,4]. However, Imamoto et al.'s protocol [14] also suffers from the same attack against Micali's schemes, while Ateniese et al.'s scheme [4] does not provide NRO and is inefficient, due to the usage of a relatively complex non-interactive knowledge proof tool called *verifiably encrypted signatures* (VES) [5,3].

We summarize the above discussion as follows. First, except the MK scheme [21], all the above mentioned schemes are *generic* constructions. That is, the two involved parties can use different types of signature schemes to generate non-repudiation evidences. In the MK scheme, however, the sender and receiver are supposed to exploit the GPS digital signature with the same parameters, since the construction is based on a special primitives called *committed signatures*, where a standard signature can be generated by two parties as in the scenario of multisignatures. Second, the property of transparent TTP is met *only* by the protocols in [21,4,14,23]. Finally, all those four schemes with transparent TTP have some limitations or security weaknesses (check Table 1 for more details). Therefore, it is desirable to propose a new generic fair non-repudiation protocol with transparent TTP to avoid those shortcomings.

1.3. Our Contribution

In this paper, we propose an efficient and secure generic fair non-repudiation protocol with transparent off-line TTP, which overcomes the limitations and security weaknesses in existing schemes. Specifically, the proposed protocol satisfies the following six desirable properties.

(1) **Generic Construction**: Each involved party can *independently* exploit *any* (secure) standard signature scheme to generate non-repudiation evidences. In particular, two involved parties are not required to use the same signature scheme.
(2) **Transparent TTP**: The generated non-repudiation evidences are the same regardless of whether the TTP is involved or not in the protocol execution.
(3) **Off-line TTP**: The third trusted party (TTP) is involved only in the situation where one party is cheating or the communication channel is interrupted. So it

could be expected that the TTP is asked to settle unfair situations *rarely*, due to the fact that fairness are always achieved, i.e., cheating is not beneficial to the cheater.

(4) **Fairness**: At the end of a protocol execution, *either* the sender Alice obtained the evidence of receipt (EOR) and the receiver Bob got the corresponding message as well as the evidence of origin (EOO), *or* none of them can get those items. This property implies that even a dishonest party who tries to cheat cannot get an advantage over the honest party.

(5) **Timeliness**: At any possible state in the protocol execution, each honest party can complete the protocol *unilaterally*, i.e., without any cooperation of the other (potentially malicious) party.

(6) **High Performance**: In normal case, the fair-exchange is finished by exchanging 3 message flows and performing 6 asymmetrically cryptographic operations. This implies our solution is the *most* efficient scheme, compared with existing schemes.

We stress that generic constructions are undoubtedly important in practice since in the real world users almost inevitably exploit different signature algorithms. At the same time, the requirement of transparent TTP is also meaningful for the following reasons. First, in a system with transparent TTP the non-repudiation evidences are simple since they are only some standard signatures generated by the involved parties. Second, the TTP is free from the burden of generating affidavits as the whole or part of non-repudiation evidences. This means the TTP's liability in our solution is further reduced, and hence the cost for the TTP's service could be cut accordingly. Third, this property is also useful to avoid bad publicity in the scenario of e-commerce. This is because the intervention of the TTP perhaps results from the communication failure instead of a party's dishonest behavior, as pointed out in [21].

1.4. Organization

The rest of the paper is organized as follows. In Section 2, we introduce assumptions and notations. Then, Section 3 presents the novel protocol in detail. After that, we discuss the security of our protocol in Section 4, and point out some extensions in Section 5. Finally, we compare our scheme with existing ones in Section 6, and conclude the paper in Section 7.

2. Assumptions and Notations

As usual, we assume that the TTP is linked with Alice and Bob by *resilient* communication channels, i.e., messages inserted into such a channel will be eventually delivered to the recipient after a finite but unknown delay. However, the communication channel between Alice and Bob may be *unreliable*, i.e., messages inserted into such a channel may be lost.

In our system, we assume that each party has a unique identifier. The identities of the sender Alice, the receiver Bob, and the TTP, are denoted as A, B, and T, respectively. We suppose that Alice, Bob and the TTP can all sign messages using (any) secure digital signature schemes, which are existentially unforgeable against an adaptive chosen message attack as defined by Goldwasser et al. in [18]. Party X's signature on a message

m is denoted as $S_X(m)$, which can be validated by using the publicly known signature verification key of party X.

Furthermore, let $(E_T(\cdot), D_T(\cdot))$ be the TTP's encryption/decryption algorithm pair, which is CCA2 secure, i.e., secure against adaptive chosen ciphertext attack as defined by Dolev et al. in [12]. To emphasize a random number R is utilized to encrypt message m, we write $c = E_T^R(m)$. Note that with the pair (m, R), anybody can verify whether a string c is the ciphertext of m with respect to the TTP's public key. We also explicitly assume (as it is generally true) that, from the ciphertext c, the TTP can recover not only m but also R. For simplicity, we call this property *randomness recoverability* and denote this fact as $(m, R) = D_T(c)$. For example, this property is satisfied by the OAEP series of encryption schemes [7,25] (especially the OAEP-RSA [13], refer to Appendix A), but not by the Cramer-Shoup cryptosystem [10].

In addition, $(E_K(\cdot), D_K(\cdot))$ denotes a pair of secure symmetric encryption and decryption algorithms with respect to secret key K, such as AES in CBC mode. When Alice wants to deliver message M to the recipient Bob, M will be encrypted as $C = D_K(M)$. Finally, let $H(\cdot)$ be a cryptographically secure one-way hash function. More notations are listed as follows.

- M: message delivered to the receiver Bob by the sender Alice.
- K: secret key used to encrypt message M.
- $f_K, f_{EOO}, f_{EOR}, f_{AT}, f_{Rec}$: publicly known *unique* flags that indicate distinct purposes of different messages in our protocol.
- $L = H(A, B, T, HC, HK)$: unique label to identify a protocol instance. That is, label L means that Alice sends message M to Bob (with/without the TTP's help), where M is determined by a ciphertext C and a symmetric key K such that $M = D_K(C)$, $HC = H(C)$ and $HK = H(K)$.
- $EK = E_T^R(f_K, L, K)$: encrypted secret key, which is the ciphertext of (f_K, L, K) under the public encryption key of the TTP by selecting a random number R.
- $EOO = S_A(f_{EOO}, L, EK)$: evidence of origin, showing that Alice sent a message M to Bob, if both EOO and EK are valid.
- $EOR = S_B(f_{EOR}, L, EK)$: evidence of receipt, showing that Bob received a message M from Alice, if both EOR and EK are valid.
- $AT = S_A(f_{AT}, L)$: abort token issued by Alice to cancel the protocol run indexed by label L.
- $Rec = S_B(f_{Rec}, L, EK)$: recover request from Bob to resolve the protocol run indexed by label L.

3. The Proposed Fair Non-Repudiation Protocol

Briefly, the basic idea of our protocol can be explained as follows. Alice first sends Bob (C, EK, EOO), where the non-repudiation of origin (NRO) is defined by the concatenation of EOO and (K, R), i.e., the correct content of EK. So, to get the committed pair (K, R), Bob has to submit his signature EOR to Alice or the TTP. At the same time, if Alice prepared EK improperly, the corresponding EOR cannot be interpreted as a valid non-repudiation evidence of receipt. This idea is partially inspired by the protocols proposed in [23,6,20]. The challenge is that many subtle problems need to be dealt properly, due to the well-known fact that security protocols are notoriously prone to err.

Like most non-repudiation protocols, our scheme consists of a dispute resolution policy, and three sub-protocols, i.e., the exchange protocol, the abort protocol, and the recovery protocol. The *dispute resolution policy* defines the evidences for non-repudiation of origin (NRO) and non-repudiation of reception (NRR), and the procedures how a judge settles potential disputes over NRO or NRR between different parties. The *exchange protocol* is the main protocol, which is the unique protocol executed jointly by the sender Alice and the receiver Bob in the *normal situation*, i.e., both involved parties behave honestly according to the protocol specification and the communication channel is in order. However, in abnormal situations, the *abort protocol* and the *recovery protocol* are further run to achieve fairness under the help of the TTP for the sender Alice and the receiver Bob, respectively. Specifically, the abort protocol allows the sender Alice to cancel a protocol instance in the following situations: (1) Bob does not respond; (2) Bob does not respond correctly or timely; or (3) The communication channel interrupts. Similarly, the recovery protocol protects the receiver Bob from the cheating of Alice or the failure of communications.

3.1. The Exchange Protocol

Assume that Alice wants to deliver a message M to the receiver Bob with the guarantee that Bob can access the message M if and only if she obtains a receipt from Bob. To this end, the sender Alice and the receiver Bob run the following exchange protocol jointly.

(e1).	$A \longrightarrow B$:	$A, B, T, C, HK, EK, EOO = S_A(f_{EOO}, L, EK)$
	`if` B gives up `then` quits	
(e2).	$B \longrightarrow A$:	$EOR = S_B(f_{EOR}, L, EK)$
	`if` A gives up `then` runs the abort protocol	
(e3).	$A \longrightarrow B$:	K, R
	`if` B gives up `then` runs the recovery protocol	

Figure 1. The Exchange Protocol

We now further explain the above protocol in detail as follows. Firstly, Alice chooses a session key K and a random number R uniformly, then computes $C = E_K(M)$, $HK = H(K)$, $HC = H(C)$, $L = H(A, B, T, HC, HK)$, $EK = E_T^R(f_K, L, K)$, and $EOO = S_A(f_{EOO}, L, EK)$. Then, Alice sends message flow (e1) to Bob. Upon receiving (e1), Bob first checks whether (A, B, T) are correct identities. If yes, he sets label $L = H(A, B, T, H(C), HK)$, and verifies whether EOO is Alice's valid signature on message (f_{EOO}, L, EK). If this is not the fact or he would not like to respond Alice, Bob just quits this protocol instance without any liability. If EOO is indeed valid, Bob could reply Alice by sending his signature $S_B(f_{EOR}, L, EK)$ as EOR in step (e2), since he is convinced that (1) if EK is correctly prepared, either Alice or the TTP can reveal (K, R) to him; and (2) if EK is incorrectly prepared, EOR is not a valid NRR evidence and so useless for Alice (and anybody else).

When message flow (e2) is received, Alice checks whether EOR is Bob's signature on message (f_{EOR}, L, EK). If this is true, Alice has gotten the non-repudiation of receipt evidence from Bob, i.e., NRR= (A, B, T, M, K, R, EOR). Thus, she reveals the values of (K, R) as message flow (e3). However, if Alice only received incorrect EOR or does not receive the EOR timely, she can run the abort protocol (see Section 3.2) to cancel this protocol instance. Note that if the confidentiality of message M is required, Alice can simply encrypt (K, R) by using Bob's public key, and then sends Bob the ciphertext instead of (K, R) itself as message flow (e3). In this way, only Bob can derive the message M, since an eavesdropper cannot obtain the symmetric key K though the encrypted message C could be intercepted.

After message flow (e3) is arrived, Bob checks whether $EK \equiv E_T^R(f_K, L, K)$. If this equality holds, Bob has obtained the non-repudiation of origin evidence from Alice, i.e., NRO$=$ (A, B, T, M, K, R, EOO). On the other hand, if (K, R) is incorrect or Bob does not receive message flow (e3) at all, he can ask for the TTP's help by initiating the recovery protocol (see Section 3.3).

3.2. The Abort Protocol

After delivering message flow (e1), if the sender Alice does not receive the expected value of EOR from Bob correctly or timely, she can execute the following abort protocol with the TTP and then cancel the protocol instance with the receiver Bob.

```
(a1).   $A \longrightarrow T:$   $A, B, T, HC, HK, E_T(AT = S_A(f_{AT}, L))$
        if (abort token $AT$ is invalid) then stop
        if (state=recovered) then retrieve $EOR$
            $T \longrightarrow A:$   $EOR$
        if (state=aborted) then retrieve confirm
            $T \longrightarrow A:$   $L$, confirm
        else set (state=aborted)
(a2).   $T \longrightarrow A:$   $L$, confirm
(a3).   $T \longrightarrow B:$   $A, B, T, HC, HK, AT = S_A(f_{AT}, L)$
```

Figure 2. The Abort Protocol

To cancel a protocol run indexed by label L, Alice first produces abort-token $AT = S_A(f_{AT}, L)$, computes $E_T(AT)$, and then sends message flow (a1) to the TTP. Upon receiving the abort request (a1), the TTP sets $L = H(A, B, T, HC, HK)$, decrypts $E_T(AT)$, and then checks whether the resulting plaintext is Alice's signature on message (f_{AT}, L). If this is not the fact, the TTP ignores the request. If AT is valid, the TTP checks whether the label L is recorded in its database. If this is true, it knows this protocol instance has been aborted or recovered, so the TTP just retrieves related items and then forwards them to Alice. Otherwise, the TTP records (L,`state=aborted`,AT, `confirm`) in its database, and then sends the message (A, B, T, HC, HK, AT) to Bob, and (L,`confirm`) to Alice by showing that her abort request is accepted. Note that in the message flow (a1) the abort-token AT is transferred in ciphertext. The purpose is to prevent Bob from intercepting AT before the TTP accepts Alice's abort request.

3.3. The Recovery Protocol

Similarly, if the receiver Bob has sent EOR to the sender Alice but does not receive the expected (K, R) from Alice correctly or timely, he can run the following recovery protocol with the TTP, and then get the values of (K, R) alternatively, if EK is indeed valid.

```
(r1).  $B \longrightarrow T:$   $A, B, T, HC, HK, EK, EOO, EOR, Rec = S_B(f_{Rec}, L, EK)$
       if any of signatures $EOO, EOR, Rec$ is invalid then stop
       if (state=aborted) then retrieve $AT$
           $T \longrightarrow B:$   $AT$
       if (state=recovered) then retrieve $(K, R)$
           $T \longrightarrow B:$   $K, R$
       if $EK$ is invalid then send an error message to Bob
       else set (state=recovered)
(r2).  $T \longrightarrow A:$   $EOR$
(r3).  $T \longrightarrow B:$   $K, R$
```

Figure 3. The Recovery Protocol

When the TTP receives a recovery request message flow (r1) from Bob, it first sets $L = H(A, B, T, HC, HK)$, and then checks whether EOO, EOR and Rec are all correct signatures on the expected messages. If this is not true, the TTP ignores the request. Otherwise, the TTP further checks whether the label L is recorded in its database. If this is true, it means this protocol instance has been aborted or recovered successfully, so the TTP just retrieves the related item and then forwards it to Bob. If L is not recorded, the TTP needs to determine the validity of EK. EK is called *valid* if and only if (1) $(f_K, L, K, R) = D_T(EK)$, where K is a symmetric key and R is a random number; and (2) $H(K) \equiv HK$. That is, a valid EK is the ciphertext of (f_K, L, K) with respect to a random number R, where the symmetric key K is consistently committed by HK. If EK is invalid [2], the TTP sends Bob an error message to tell him this fact. Otherwise, i.e., EK is indeed valid, the TTP records (L,state=recovered,Rec,EOR,(K, R)) in its database, and sends EOR to Alice and (K, R) to Bob. In addition, to provide the confidentiality of message M the TTP can deliver (K, R) to Bob in ciphtertext, as we mentioned in the exchange protocol.

3.4. The Dispute Resolution Policy

In some day after the completion of a protocol execution (with or without the TTP's participation), a judge may need to settle the following two types of dispute.

- **Repudiation of Origin**. If Alice denies having sent message M to Bob, Bob could send NRO=(A, B, T, M, K, R, EOO) to a judge as a dispute resolution request. Then, the judge performs as follows:
  - (1) Check that (A, B, T) are identifies of Alice, Bob, and a TTP who provides service for fair non-repudiation exchange.

[2]Namely, EK is *invalid* in all of the following cases: (a) The TTP cannot decrypt EK, i.e., EK is an invalid ciphertext; (b) The derived plaintext of EK is not equivalent to the expected tuple (f_K, L, K, R), i.e., the concatenation of two known values (f_K, L) and two unknown values (K, R); (c) $H(K) \neq HK$, i.e., the derived key K is inconsistent with the committed key HK.

(2) Compute $C = E_K(M)$, $L = H(A, B, T, H(C), H(K))$ and $EK = E_T^R(f_K, L, K)$.
(3) Check whether EOO is Alice's valid signature on message (f_{EOO}, L, EK).
(4) Accept Bob's claim if all above checks hold. Otherwise, reject Bob's claim.

- **Repudiation of Receipt**. Similarly, if Bob denies having received message M from Alice, Alice could provide NRR$= (A, B, T, M, K, R, EOR)$ to a judge for dispute resolution. Then, the judge acts in the following way:

  (1) Check that (A, B, T) are identifies of Alice, Bob, and a TTP who provides service for fair non-repudiation exchange.
  (2) Compute $C = E_K(M)$, $L = H(A, B, T, H(C), H(K))$ and $EK = E_T^R(f_K, L, K)$.
  (3) Check whether EOR is Bob's valid signature on message (f_{EOR}, L, EK).
  (4) If any of the above checks fails, reject Alice's claim.
  (5) Enquire whether Bob (or the TTP) has abort token AT. If a valid AT is provided, reject Alice's claim. Otherwise, Alice's claim is accepted.

Note that when solving the repudiation of receipt, the judge is required to check whether Bob (or the TTP) holds abort token AT. The reason is that Alice may get valid EOR as well as successfully abort a protocol run in the following two scenarios: (a) After she aborted a protocol run, (honest) Alice gets valid EOR from Bob due to the communication delay; (b) After valid EOR has been received, dishonest Alice promptly launches the abort protocol. By doing so, Alice has gotten NRR, but Bob can only get AT, instead of (K, R), from the TTP. Hence, valid AT should counteract the power of correct but illegal EOR. However, it is a different story in the procedure of repudiation of origin, because valid NRO and AT only imply Alice is trying to cheat: She revealed (K, R) to Bob as well as executed the aborted protocol! Therefore, in this case valid NRO is considered as the proof that Alice has delivered message M to Bob, regardless of whether Alice holds valid AT or not.

4. Security Discussions

Based on previous description and discussion, we know that in the normal situation, i.e., both involved parties are honest and the communication channel is in order, the sender will receive valid NRR, while the receiver will get the message M and valid NRO, and the TTP is not involved. In other words, our scheme is *complete* and *optimistic*. *Timeliness* is also satisfied, since both the sender and the receiver can terminate a protocol instance unilaterally by initiating the abort protocol or recovery protocol, respectively. In addition, it is obvious that our protocol is a *generic* scheme and supports *transparent* TTP.

Now, we discuss the most important security requirement for a fair exchange protocol: fairness. That is, we have to show that in our scheme, any of the two involved parties cannot take advantage over the other in the end of any possible protocol execution, even if he or she behaves dishonestly. We classify our discussion into two cases: (1) Alice is honest, but Bob is trying to cheat; and (2) Bob is honest, but Alice is trying to cheat.

Case 1: *Alice is honest, but Bob is trying to cheat*. Since Alice is honest now, the whole message flow (e1) delivered to Bob is correctly prepared. Therefore, when message flow (e1) is received, Bob will find EOO is indeed Alice's valid signature on message (f_{EOO}, L, EK), where the label $L = H(A, B, T, H(C), HK)$. After that, Bob has to make a choice whether he wants to return his signature EOR to Alice. If Bob does, honest initiator Alice will reveal the correct values of (K, R) as Bob expects. In such situation, Bob gets valid NRO as well as the message M from Alice, while Alice also obtains valid NRR from Bob simultaneously. So the protocol execution is fair. If Bob does not send EOR or only sends an incorrect EOR to Alice, he cannot get correct (K, R) from Alice via message flow (e3). However, EK is a ciphertext produced by Alice under the TTP's public encryption algorithm, which is CCA2 secure. So, except Alice, only the TTP can derive (K, R) from the ciphertext EK. This means that the TTP is Bob's last resort to get (K, R) by running the recovery protocol. To do this, there are only two strategies.

The first one is to send the TTP all correct items in message flow (r1), including EK, EOO, EOR ect. In this case, if Alice aborted the protocol run indexed by the label L, Bob can only get a valid AT. If this protocol run is not aborted yet, Bob can successfully get the value of (K, R) from the TTP, but Alice will obtain valid EOR too. Another possible strategy is to initiate the recovery protocol by using some new items in message flow (r1). However, to get (K, R) from EK and keep the consistency between EK and HK, Bob has to submit original HK and EK. In other words, Bob can only submit the TTP a message flow (r1') with the following format: $A', B', T, HC', HK, EK, EOO_{A'}, EOR_{B'}, Rec_{B'}$, where A' and B' are some parties colluding with Bob. But this attack is useless again, since the TTP will find the new label $L' = H(A', B', T, HC', HK)$ does not equal to the label L committed in EK (except a negligible probability), due to the assumption that $H(\cdot)$ is a secure one-way hash function. Therefore, even if Bob asks for the TTP's help in a cheating way, our protocol is still fair for honest sender Alice.

Case 2: *Bob is honest, but Alice is trying to cheat.* The purpose of dishonest sender Alice is to get valid NRR from Bob such that Bob cannot access the message M or does not receive valid NRO. In our protocol, Alice may dishonestly execute any or some of the following steps: (e1), (e3), and the abort protocol. In message flow (e1), the identities (A, B, T) and EOO should be valid. Otherwise, according to the specification of our exchange protocol, honest Bob will not respond Alice at all. So, it seems Alice can send out incorrect information (even random strings) for the following items: C, HK, and EK. However, C and HK determines the message M by $M = D_K(C)$ and $H(K) = K$. So, if those two items are incorrect, both EOR and EOO cannot be interpreted as valid NRR and NRO. Furthermore, if Alice prepared EK incorrectly, EOR is also useless for anybody (including Alice), since EOR includes both EK and the label L, while EK includes L again. At the same time, the symmetric key committed by both HK and EK should be consistent. Otherwise, both EOO and EOR are useless again. Therefore, we conclude that to get valid EOR from honest Bob, Alice has to correctly prepare message flow (e1).

Another cheating strategy is to get valid EOR first and then refuse to reveal (K, R) and/or run the abort protocol. Just refusing to reveal (K, R) does not harm Bob in essence, since he can get correct (K, R) from the TTP by executing the recovery protocol. On the other hand, if Alice not only refuses to reveal (K, R) but also initiates the abort protocol, Bob will get valid abort token AT from the TTP though Bob indeed cannot get the values of (K, R). However, according to the specification of our dispute resolution policy, when repudiation of receipt occurs Bob can provide this valid AT to invalidate EOR provided by Alice. Hence, our protocol is fair for honest Bob too.

Based on the above analysis, we conclude that in the proposed protocol a dishonest party cannot take advantage over the other honest party. In other words, our protocol satisfies the property of *fairness*.

5. Extensions

In this section, we briefly discuss some possible extensions of the proposed protocol. Firstly, we implicitly assume that in our protocol the TTP always stores all the state information in its searching database. Especially, it is necessary to correctly record whether a protocol instance indexed by a label L has been aborted or recovered. Otherwise, the TTP may mistakenly confirm Alice's abort request as well as accept Bob's recovery request. In practice, the TTP's storage is limited. To reduce such data in the TTP's searching database, we can introduce a time limit t to our protocol. For example, define $L = H(A, B, T, HC, HK, t)$, where t indicates both the beginning time t_1 and the end time t_2. That is, both the abort protocol and the recovery protocol cannot be executed before time t_1 or after time t_2. After t_2 expires, the TTP can remove all state information related to time limit t from its searching database into a log system or just print them as hard copies. Naturally, time limit t should be agreed by both Alice and Bob, and long enough for them (e.g. 24 hours). In a concrete implementation, however, cares should be paid for the possible attacks resulting from the drift of clocks among different parties.

Secondly, another variant could be obtained by deleting the abort protocol but adding time limit t in our scheme. In other words, after Alice delivered message flow (e1) to Bob, she cannot abort this protocol instance at all, but she only needs to wait EOR for a limit time, i.e., until t_2. If Alice neither receives valid EOR from Bob nor from the TTP after time t_2, this protocol run is deemed to be cancelled since the TTP will not accept Bob's recovery request any more. This extension has twofold meanings: (1) The whole non-repudiation protocol becomes simpler; and (2) This variant archives the property of *stateless TTP*, i.e., in theory the TTP has no need to store any state information to maintain fairness. The disadvantage of this variant is that the sender Alice may need to wait up to a finite time for the termination of a protocol instance. That is, only *weak timeliness* is achieved.

Thirdly, using the techniques of threshold cryptography the proposed protocol could be extended for the scenarios where the trust on a single TTP needs to be distributed into multiple relatively less trustworthy TPPs, or a non-repudiation evidence is required to be signed only by a given quota of members in a group cooperatively.

Finally, by exploiting techniques from [15,24], it is also possible to extend our protocol as a multi-party non-repudiation scheme where one sender is able to deliver the same message (or different messages) to many recipients efficiently.

6. Comparison

Table 1 shows the comparison of basic features, security, and efficiency between our new protocol and a number of the state-of-the-art non-repudiation protocols [4,14,16,20,21,

	Generic Constr.	Transp. TTP	Off-line TTP	EOR	EOO	Fair-ness	Time-liness	♯ of Mess.	♯ of Oper.
ZG [26]	Yes	No	No	Yes	Yes	Yes*	Weak	5	9
ZG [27]	Yes	No	Yes	Yes	Yes	Yes*	Weak	4	8
ZDB [28,29]	Yes	No	Yes	Yes	Yes	Yes*	Yes	4	12
KMZ [16]	Yes	No	Yes	Yes	Yes	Yes*	Yes	4	12
GRV [20]	Yes	No	Yes	Yes	Yes	Yes	Yes	4	10
MK [21]	No	Yes	Yes	Yes	Yes	Yes*	Yes	4	12
AN [4]	Yes	Yes	Yes	Yes	No	Yes	No	4	17
Micali [23]	Yes	Yes	Yes	Yes	No	Yes*	Weak	3	8
IS [14]	Yes	Yes	Yes	Yes	Yes	Yes*	No	3	8
Ours	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**3**	**6**

23,26,27,28,29]. In Table 1, those schemes are listed in an order for easy comparison, not according to the chronology. In the category of basic features, we consider five properties: generic scheme or not, transparent TTP or not, off-line or on-line TTP, and whether both EOR and EOO evidences are provided. The result shows that only four previous schemes [21,4,23,14], as well as our new protocol, support transparent TTP. Three of those four schemes are generic constructions, while the MK scheme [21] is a specific scheme, as we mentioned in Section 1.

Here, we only compare two main security requirements: fairness and timeliness. Almost all schemes satisfy timeliness by providing both the abort and recovery protocols, while the ZG schemes [26,27] and Micali's schemes [23] meet only weak timeliness due to the usage of a deadline. As we discussed above, using deadline is an interesting method to achieve stateless TTP. Except the two schemes proposed in [4,20], fairness in other schemes is affected by the attacks (in different flavors) identified by Gürgens et al.'s [19,20] and us. Fortunately, it seems that all those schemes could be repaired by more or less modifications, though the security of revised schemes should be checked carefully again. Due to this reason, we mark those schemes with "Yes*" under the column of "fairness".

In the efficiency evaluation, we compare the costs of both communication and computation in the *normal* case. In other words, the overloads of the abort and recovery protocols are not discussed here, since those events are supposed to occur abnormally and rarely. To complete a successful exchange, only Micali's schemes, the IS protocol, and our solution need to transfer 3 messages between the two involved parties, while this number is 4 or 5 in other schemes. In the exchange protocol of our scheme, Alice and Bob are required to perform the following main computations: (a) Encrypt and verify the encrypted symmetric key EK under the TTP's public key; and (b) Sign and verify two signatures, i.e., EOO and EOR. Therefore, our protocol needs 6 asymmetrical cryptographic operations. However, this number representing computation cost varies from 8 to 17 in other schemes. Note that we do not count into the computation cost of symmetric encryption and decryption, i.e., $C = E_K(M)$ and $M = D_K(C)$, since symmetrical operations are much faster than asymmetric operations, especially for common messages (not very long).

7. Conclusion

In this paper, we first briefly reviewed a number of non-repudiation protocols and certified e-mail schemes. In particular, we identified a potential attack on the schemes in [14,23], where a sender can cheat a receiver by mixing identities of different TTPs when they execute the non-repudiation protocol. Consequently, we concluded that to overcome limitations and security shortcomings in previous schemes, it is desirable in practice to propose a new *generic* fair non-repudiation protocol with *transparent* off-line TTP. To this end, we proposed such a new fair non-repudiation protocol, and pointed out some possible extensions. Compared with the existing solutions, our protocol is not only secure but the *most* efficient. Actually, this new protocol is also the *first* three-move fair non-repudiation protocol supporting timeliness and transparent off-line TTP. In the future work, we will consider to exploit formal tools [17,22] to analyze the proposed non-repudiation protocol.

References

[1] N. Asokan, M. Schunter, and M. Waidner. Optimistic protocols for fair exchange. In: *Proc. of AMC Conference on Computer and Communications Security* (*CCS'97*), pp. 7-17. ACM Press, 1997.

[2] N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. *IEEE Journal on Selected Areas in Communications*, 18 (4): 591-606, 2000.

[3] G. Ateniese. Efficient verifiable encryption (and fair exchange) of digital signature. In: *Proc. of AMC Conference on Computer and Communications Security* (*CCS'99*), pp. 138-146. ACM Press, 1999.

[4] G. Ateniese and C. Nita-Rotaru. Stateless-receipient certified E-mail system based on verifiable encryption. In: *CT-RSA'02*, LNCS 2271, pp. 182-199. Springer-Verlag, 2002.

[5] F. Bao, R.H. Deng, and W. Mao. Efficient and practical fair exchange protocols with off-line TTP. In: *Proc. of IEEE Symposium on Security and Privacy*, pp. 77-85. IEEE Computer Society, 1998.

[6] F. Bao, G. Wang, J. Zhou, and H. Zhu. Analysis and improvement of Micali's fair contract signing protocol. In: *Information Security and Privacy* (*ACISP'04*), LNCS 3108, pp. 176-187. Springer-Verlag, 2004.

[7] M. Bellare and P. Rogaway. Optimal asymmetric encryption - How to encrypt with RSA. In: *EUROCRYPT'94*, LNCS 950, pp. 92-111. Springer-Verlag, 1994.

[8] C. Boyd and E. Foo. Off-line fair payment protocols using convertible signatures. In: *ASIACRYPT'98*, LNCS 1514, pp. 271-285. Springer-Verlag, 1998.

[9] T. Coffey and P. Saidha. Non-repudiation with mandatory proof receipt. *Computer Communication Review*, 26(1): 6-17, 1996.

[10] R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: *CRYPTO'98*, LNCS 1462, pp.13-25. Springer-Verlag, 1998.

[11] I.B. Damgård. Practical and provably secure release of a secret and exchange of signatures. *Journal of Cryptology*, 8(4): 201-222, 1995.

[12] D. Dolev, D. Dwork, and N. Naor. Non-mealleable cryptography. In: *SIAM Journal on Computing*, 2000, 30(2): 391-437.

[13] E. Fujisaki, T. Okamoto, D. Pointcheval, and J. Stern. RSA-OAEP is secure under the RSA assumption. In: *CRYPTO'01*, LNCS 2139, pp. 260-274. Springer-Verlag, 2001.

[14] K. Imamoto and K. Sakurai. A cerified e-mail system with receiver's selective usage of delivery authortiy. In: *Indocrypt'02*, LNCS 2551, pp. 326-338. Springer-Verlag, 2002.

[15] S. Kremer and O. Markowitch. A multi-party non-repudiation protocol. In: *Information Security for Global Information Infrastructures* (*IFIP/SEC'00*), pp. 271-280. Kluwer, 2000.
[16] S. Kremer, O. Markowitch, and J. Zhou. An intensive survey of fair non-repudiation protocols. *Computer Communications*, 25(17): 1606-1621, Nov. 2002.
[17] S. Kremer and J.-F. Raskin. A game-based verification of non-repudiation and fair exchange protocols. *Journal of Computer Security*, 11(3): 399-430, 2003.
[18] S. Goldwasser, S. Micali, and R. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, April 1988, 17(2): 281-308.
[19] S. Gürgens and C. Rudolph. Security analysis of (un-) fair nonrepudiation protocols. In: *Formal Aspects of Security* (*FASec'02*), LNCS 2629, pp. 97-114. Springer-Verlag, 2003.
[20] S. Gürgens, C. Rudolph, and H. Vogt. On the security of fair non-repudiation protocols. In: *Information Security Conference* (*ISC'03*), LNCS 2851, pp. 193-207. Springer-Verlag, 2003.
[21] O. Markowitch and S. Kremer. An optimistic non-repudiation protocol with transparent trusted third party. In: *Information Security Conference* (*ISC'01*), LNCS 2200, pp. 363-378. Springer-Verlag, 2001.
[22] C. Meadows. Formal methods for cryptographic protocol analysis: Emerging issues and trends. *IEEE Journal on Selected Areas in Communications*, 21(1): 44-54, Jan. 2003.
[23] S. Micali. Simple and fast optimistic protocols for fair electronic exchange. In: *Proc. of 22th Annual ACM Symp. on Principles of Distributed Computing* (*PODC'03*), pp. 12-19. ACM Press, 2003.
[24] J. A. Onieva, J. Zhou, M. Carbonell, and J. Lopez. A multi-party non-repudiation protocol for exchange of different messages. In: *Security and Privacy in the Age of Uncertainty* (*IFIP/SEC'03*), pp. 37-48. Kluwer, 2003.
[25] V. Shoup. OAEP reconsidered. *Journal of Cryptology*, 15(4): 223-249, 2002.
[26] J. Zhou and D. Gollmann. A fair non-repudiation protocol. In: *Proc. of the IEEE Symposium on Security and Privacy*, pp. 55-61. IEEE Computer Society Press, 1996.
[27] J. Zhou and D. Gollmann. An efficient non-repudiation protocol. In: *Proc. of the 10th Computer Security Foundations Workshop* (*CSFW'97*), pp. 126-132. IEEE Computer Society Press, 1997.
[28] J. Zhou, R. Deng, and F. Bao. Evolution of fair non-repudiation with TTP. In: *Information Security and Privacy* (*ACISP'99*), LNCS 1587, pp. 258-269. Springer-Verlag, 1999.
[29] J. Zhou, R. Deng, and F. Bao. Some remarks on a fair exchange protocol. In: *Public Key Cryptography* (*PKC'00*), LNCS 1751, pp. 46-57. Springer-Verlag, 2000.

Appendix A: Review of OAEP

The Optimal Asymmetric Encryption Padding (OAEP) is first introduced by Bellare and Rogaway in [7], which is used to convert a trapdoor permutation to a public key encryption cryptosystem with semantic security against adaptive chosen-ciphertext attacks, i.e., CCA2 security [12]. Fujisaki et al. [13] formally proved that OAEP is CCA2 secure in the random oracle model under the partial-domain one-wayness of the underlying permutation. Shoup proposed a slightly modified version of OAEP, called OAEP+, which is provably secure under the one-wayness of the permutation (weaker than Fujisaki et al.'s assumption). However, since partial-domain one-wayness of the RSA function is equivalent to the (fulldomain) one-wayness, the security of RSA-OAEP can actually be proven under the one-wayness of the RSA function.

Let k_0, k_1, k and n be proper security parameters, such that $n = k - k_0 - k_1$. In addition, G and H denote two hash functions:

$$G : \{0,1\}^{k_0} \longrightarrow \{0,1\}^{k-k_0} \quad \text{and} \quad H : \{0,1\}^{k-k_0} \longrightarrow \{0,1\}^{k_0}.$$

The following is the description of the OAEP cryptosystem $(\mathcal{K}, \mathcal{E}, \mathcal{D})$ obtained from a permutation f, whose inverse is denoted by g.

- **Key generation algorithm** $\mathcal{K}(1^k)$: given the security parameter k, probabilistically outputs an instance of the function f with its inverse g. Then, sets f as the public key pk, and g as the private key sk.
- **Encryption algorithm** $\mathcal{E}_{pk}(m; r)$: for a given message $m \in \{0,1\}^n$, selects a random number $r \in \{0,1\}^{k_0}$, then computes

$$s = (m||0^{k_1}) \oplus G(r) \quad \text{and} \quad t = r \oplus H(s),$$

  and outputs the ciphertext $c = f(s,t)$.
- **Decryption algorithm** $\mathcal{D}_{sk}(c)$: given ciphertext c, first using the private $sk = g$ extracts

$$(s,t) = g(c), \quad r = t \oplus H(s), \quad \text{and} \quad M = s \oplus G(r).$$

  Then, if $[M]_{k_1} \equiv 0^{k_1}$, the algorithm outputs plaintext $[M]^n$; otherwise it returns "reject". Here, $[M]_{k_1}$ denotes the k_1 least significant bits of M, while $[M]^n$ denotes the n most significant bits of M.

According to the above specification, it is obvious that OAEP series cryptosystems [7,25] satisfy the *randomness recoverability*, i.e., the random number r can be recovered in the decryption procedure. Therefore, in our protocol, if Alice refuses to reveal the pair (K, R) to Bob, the TTP can decrypt the encrypted secret key EK and then forward both the secret key K and the random number R to Bob. At the same time, the values of (K, R) allows any third party to verify whether the alleged encrypted key EK correctly corresponds to the message (f_K, L, K) under the TTP's public key by deterministically checking $EK \equiv E_T^R(f_K, L, K)$. That is what we need in our fair non-repudiation protocol.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Efficient Authorization in Delegation Chains with Strong Non-Repudiation

Richard W.C.Lui [1], S.M. Yiu and Lucas C.K.Hui
Department of Computer Science,
The University of Hong Kong,
Pokfulam, Hong Kong

Abstract. In an organization, it is a common practice for a user (the delegator) to delegate some rights, in particular the signing right, to another user (the delegate). From the perspective of digital signature, a secure scheme is required to handle the delegation process so that the authorization as well as the signature of the delegate can be verified efficiently. In general, delegation can occur more than one level, thus forming a delegation chain. Among the existing approaches, delegation certificate [1] is a popular technique for performing delegation and handling chained delegation. However, it is not scalable because the verification of authorization is inefficient.

In this paper, we extend Kim et al.'s proxy signature [6], which only handles one level of delegation, to support efficient verification for a delegation chain. We first show that a straight-forward extension of Kim et al.'s scheme does not support strong non-repudiation. We propose a possible way to modify the scheme to support the property.

Keywords. Non-repudiation, Delegation, Proxy Signature

1. Introduction

Delegation is commonly used in organizations to delegate some permission by one user to another user to achieve organization goals [9]. The scenario we are interested in this paper concerns the chained delegation of signing rights. A user (the original delegator) performs delegation of signing right to another user (the intermediate delegator), who re-delegates. This delegation process can continue to build up a delegation chain. Any delegate (the proxy signer) in the chain can sign a message on behalf of the original delegator. From the perspective of digital signatures, a scheme is required for handling the delegation process so that the signature and the authorization of the delegate can be verified. There are two common approaches for solving this problem, the delegation certificates and the proxy signatures.

Delegation certificates: Delegation certificates [1] require the users to hold multiple certificates, one for each level of delegation. So, to verify a signature signed by a delegate, the verifier has to perform a series of verification operations, one for each delegation cer-

[1]Correspondence to: Richard W.C.Lui, Department of Computer Science, The University of Hong Kong, Pokfulam, Hong Kong. Tel.: +852 28578263; E-mail: wclui@cs.hku.hk

tificate and also one for the signature of the delegate. In other words, when the delegation chain is long, it becomes quite inefficient and inconvenient.

Proxy signatures: The notion of proxy signature [8,6,7] is introduced by Mambo et al. [8]. Various security requirements such as verifiability, strong unforgeability, strong identifiability and strong undeniable are introduced. However, the proposed schemes support only unlimited delegation to the proxy signer. Kim et al. [6] extends it by using Schnorr signature and delegation warrant. In this model, the original delegator can specify, in the delegation warrant, that he/she will delegate only part of his/her signing right to the proxy signer. With a proxy signature, the verifier can check the delegation warrant and determine whether the proxy signer has the required right to perform proxy signing. By using proxy signature, extra verification of the delegation certificates in a delegation chain is not needed and so the process of signature verification is more efficient.

Kim et al.'s scheme has been proved to be secure if the underlying signature scheme (i.e., Schnorr signature scheme) is secure [2]. However, that scheme only handles one level of delegation. Ding and Petersen [3] then proposed a scheme to handle a delegation chain using proxy signatures. However, in their scheme, the user's private key is known by the certificate authority (CA). Therefore, non-repudiation cannot be supported.

Our contributions and organization of the paper: In this paper, we introduce the notion of strong non-repudiation in chained delegation of signing rights. We show that a straight-forward extension of the Kim et al.'s scheme is subject to a new form of attack from an insider and does not support strong non-repudiation. Based on our observation, we show how to extend the scheme in a secure way to handle authorization in delegation chains. The efficiency and strong non-repudiation property of the scheme is studied.

The rest of the paper is organized as follows. We review Kim et al.'s scheme in Section 2. In Section 3, the strong non-repudiation property in chained delegation is introduced and we show that a straight-forward extension of the Kim et al.'s scheme does not support strong non-repudiation. We propose an efficient scheme which supports strong non-repudiation in chained delegation. We compare our proposed delegation scheme with a related scheme [3], which also handles chained delegation of signing rights using proxy signature techniques. Finally, we conclude our paper in Section 4.

2. Review of the Proxy Signature Scheme Proposed by Kim et al. [6]

In this section, we outline Kim et al.'s proxy signature scheme. The scheme consists of the proxy issuing and the proxy signing protocol. Let p and q be large primes such that q divides $p-1$. Let g be a generator of a multiplicative subgroup of Z_p^* with order q, $h()$ denotes a collision resistant cryptographic hash function with range Z_q, $(x, y = g^x)$ be the private and public key respectively. Suppose Alice (with a personal key pair $(x_A \in_R Z_q^*, y_A = g^{x_A} \ (mod\ p))$ intends to delegate her signing right to Bob (with a personal key pair $(x_B, y_B = g^{x_B} \ (mod\ p)))$. Alice computes the proxy for Bob by randomly generating a key pair $(k_A \in_R Z_q^*, r_A = g^{k_A} \ (mod\ p))$ and computing the proxy s_A as follows.

The proxy: $s_A = x_A\, h(w_A, r_A) + k_A \ (mod\ q)$

where w_A is the delegation warrant which specifies the public key of Alice, Bob, and the restrictions on the use of this delegation. Since the role of the original delegator and the proxy signer is not apparent in the verification relation, this relation has to be explicitly stated in the delegation warrant [7]. The proxy will be transferred to Bob. It has been shown that the proxy can be transferred without a secure channel [5]. Bob then verifies the proxy by checking if $g^{s_A} = y_A^{h(w_A,r_A)} r_A \ (mod\ p)$. After the verification, Bob can generate the private key and public key for his proxy signature as follows.

The proxy private key: $p_B = s_A + x_B\ h(w_A, r_A)\ (mod\ q)$

The proxy public key: $t_B = (y_A\ y_B)^{h(w_A,r_A)} r_A (mod\ p)$

To generate a proxy signature on a message M, Bob randomly generates a per-message key pair $(k \in_R Z_q^*, r = g^k\ (mod\ p))$ and uses Schnorr's signature scheme [10][1] to sign the message using the proxy private key p_B. It can be easily shown that by following the verification procedure of Schnorr's signature, the verifier can check the validity of the signature. Moreover, the verifier should check that the public key y_B is specified in w_A to be the delegate to make sure that the proxy is used by an authorized party.

3. The proposed delegation scheme

3.1. A Faulty Extension of Kim et al.'s Scheme

In this section, we try to extend Kim et al.'s scheme to handle a delegation chain in a straight-forward manner. The basic idea is that if Bob wants to re-delegate his signing right to Carol, Bob can use his proxy private key, p_B to form a proxy for Carol. We then show that the extension has a flaw. We demonstrate the flaw by showing an attack from an insider.

Let $C =< c_1, c_2, ..., c_n >$ be a delegation chain for $n > 1$ where user c_i delegates his/her signing right to user c_{i+1} for $i = 1, 2, \ldots, n-1$. Consider user c_i. We denote x_{c_i} and y_{c_i} to be the private and public key of the user respectively. Also, we denote w_{c_i} to be the delegation warrant and s_{c_i} to be the proxy issued by the user respectively. In addition, we denote (k_{c_i}, r_{c_i}) to be a randomly generated ephemeral key pair for that user and let $A_{c_i} = h(w_{c_i}, r_{c_i})$. Also, we let p_{c_i} to be the proxy private key, which is used by the user to discharge the delegated rights. See Figure 1 for an overview of the extension.

In general, user c_i, where $1 < i < n$, receives the proxy $s_{c_{i-1}}$ from user c_{i-1} and he/she generates the proxy s_{c_i} for user c_{i+1}. User c_{i+1} then generates the proxy private key $p_{c_{i+1}}$ and the proxy public key $t_{c_{i+1}}$. This process is carried out in a similar manner along the whole chain. The values of $s_{c_i} (i = 1, 2, \ldots, n-1)$, p_{c_j} and t_{c_j} $(j = 2, 3, \ldots, n)$ in the scheme are calculated according to the following recurrences.

Basis:

$$\begin{aligned}
s_{c_1} &= (x_{c_1} A_{c_1} + k_{c_1})\ (mod\ q)\\
p_{c_2} &= (x_{c_1} A_{c_1} + x_{c_2} A_{c_1} + k_{c_1})\ (mod\ q)\\
t_{c_2} &= ((y_{c_1} y_{c_2})^{A_{c_1}} r_{c_1})\ (mod\ p)
\end{aligned}$$

Recurrence (for $i = 2, 3, \ldots, n-1$):

[1]One can use ElGamal's signature scheme [4] to replace Schnorr's signature scheme.

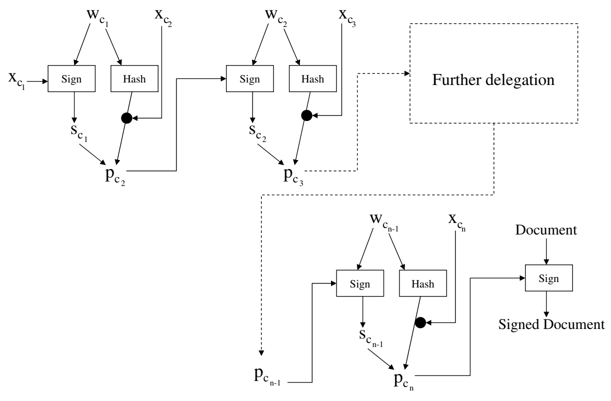


Figure 1. The faulty scheme for chained delegation of signing rights (User $c_1, c_2, ..., c_n$)

$$\begin{aligned}
s_{c_i} &= (p_{c_i}A_{c_i} + k_{c_i})\ (mod\ q)\\
p_{c_{i+1}} &= (p_{c_i}A_{c_i} + x_{c_{i+1}}A_{c_i} + k_{c_i})\ (mod\ q)\\
t_{c_{i+1}} &= ((t_{c_i}y_{c_{i+1}})^{A_{c_i}} r_{c_i})\ (mod\ p)
\end{aligned}$$

By expanding the recurrences, we obtain the following equations for $p_{c_{i+1}}$ and $t_{c_{i+1}}$.

Lemma 1 *The proxy s_{c_i}, the proxy private key $p_{c_{i+1}}$ and the proxy public key $t_{c_{i+1}}$ for user c_{i+1} can be expressed by the following equations.*

$$\begin{aligned}
s_{c_i} &= x_{c_1}(A_{c_1}A_{c_2}\dots A_{c_i}) + \textstyle\sum_{\sigma=2}^{i}(x_{c_\sigma}\prod_{j=\sigma-1}^{i}A_{c_j})\\
&\quad + \textstyle\sum_{\sigma=1}^{i-1}(k_{c_\sigma}\prod_{j=\sigma+1}^{i}A_{c_j}) + k_{c_i}\ (mod\ q)\\
p_{c_{i+1}} &= x_{c_1}(A_{c_1}A_{c_2}\dots A_{c_i}) + \textstyle\sum_{\sigma=2}^{i}(x_{c_\sigma}\prod_{j=\sigma-1}^{i}A_{c_j}) + x_{c_{i+1}}A_{c_i}\\
&\quad + \textstyle\sum_{\sigma=1}^{i-1}(k_{c_\sigma}\prod_{j=\sigma+1}^{i}A_{c_j}) + k_{c_i}\ (mod\ q)\\
t_{c_{i+1}} &= y_{c_1}^{A_{c_1}A_{c_2}\dots A_{c_i}}\textstyle\prod_{\sigma=2}^{i} y_{c_\sigma}^{A_{c_{\sigma-1}}A_{c_\sigma}\dots A_{c_i}}(y_{c_{i+1}})^{A_{c_i}}\prod_{\sigma=1}^{i-1} r_{c_\sigma}^{A_{c_{\sigma+1}}\dots A_{c_i}}\\
&\quad r_{c_i}\ (mod\ p)
\end{aligned}$$

Proof The results can be obtained by expanding the recurrences directly. ▌

It can be easily verified that the proxy signature created by c_n can be checked by the verification procedure provided by Schnorr's signature scheme. Moreover, the verifier should check that the public key y_{c_i} is specified to be the delegate in $w_{c_{i-1}}$ to make sure that the proxy is used by an authorized party for all $1 < i < n$.

3.2. An Attack to Manipulate the Delegation Chain

In this section, we show that there is a flaw in the above scheme. Although the delegation scheme is constructed from a proxy signature scheme which is provably secure, the delegation scheme does not support strong non-repudiation. We first define the notion of strong non-repudiation as follows. Given a proxy signature which is generated by the proxy signer using the delegated permission of a delegation chain C, he/she should not be able to repudiate that he/she generates the signature. Also, he/she should not be able

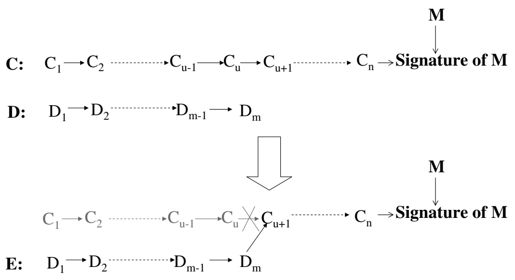

M
C: C1 → C2 → Cu-1 → Cu → Cu+1 → Cn → Signature of M
D: D1 → D2 → Dm-1 → Dm
M
C1 → C2 → Cu-1 → Cu → Cu+1 → Cn → Signature of M
E: D1 → D2 → Dm-1 → Dm

Figure 2. Manipulating the delegation chain

to falsely claim that the signature is generated on behalf of him/her, or the original delegator of another delegation chain $C' \neq C$. In addition, the original and intermediate delegators in C should not be able to repudiate that they have authorized the signing by performing delegation/re-delegation.

We consider an attack launched by user c_u, where $1 \leq u < n$, in a delegation chain $C =< c_1, ..., c_u, ..., c_n >$. The idea of the attack is as follows (see Figure 2). Suppose the attacker makes use of the rights associated with the delegation chain C to sign on behalf of user c_1 for a message M. In addition, he/she obtains a proxy $s_{d_{m-1}}$ for the delegation chain $D =< d_1, d_2, ..., d_m >$ where d_m and c_u is the same user. It is possible for the attacker to manipulate the proxy signature (on M) in a way that the delegation chain C will be substituted by the delegation chain $E =< d_1, d_2, ..., d_{m-1}, c_u, c_{u+1}, ..., c_n >$. As a result, user c_n will unknowingly perform proxy signing of M on behalf of another user d_1.

We describe how it is possible for user c_u to launch the attack. Let M be a message and let the proxy signature on M by c_n be $s_C(M) = p_{c_n} h(M, r) + k \ (mod\ q)$, $(k \in_R \ Z_q^*, r = g^k \ (mod\ p))$ is a randomly generated per-message key pair. The user c_u may convert $S_C(M)$ to another valid signature $S_E(M)$ using delegation chain E as follows.

Theorem 1 *The user c_u can compute*

$$s_E(M) = s_C(M) + (s_{d_{m-1}} + x_{c_u} A_{d_{m-1}} - (s_{c_{u-1}} + x_{c_u} A_{c_{u-1)}}))h(M, r) \prod_{i=u}^{n-1} A_{c_i} \ (mod\ q)$$

such that $s_E(M)$ is a valid proxy signature for M based on the delegation chain E.

Proof Based on Lemma 1, the values of $s_C(M)$, $s_{c_{u-1}}$ and $s_{d_{m-1}}$ can be expressed as follows.

$$\begin{aligned}
s_C(M) &= (p_{c_n} h(M,r) + k)\ (mod\ q) \\
&= ((x_{c_1}(A_{c_1} \ldots A_{c_{n-1}}) + \textstyle\sum_{\sigma=2}^{u-1} (x_{c_\sigma} \prod_{j=\sigma-1}^{n-1} A_{c_j}) \\
&\quad + x_{c_u}(A_{c_{u-1}} \ldots A_{c_{n-1}}) + \textstyle\sum_{\sigma=u+1}^{n-1} (x_{c_\sigma} \prod_{j=\sigma-1}^{n-1} A_{c_j}) + x_{c_n} A_{c_{n-1}} \\
&\quad + \textstyle\sum_{\sigma=1}^{u-2} (k_{c_\sigma} \prod_{j=\sigma+1}^{n} A_{c_j}) + \sum_{\sigma=u-1}^{n-2} (k_{c_\sigma} \prod_{j=\sigma+1}^{n-1} A_{c_{\sigma+1}}) \\
&\quad + k_{c_{n-1}}) h(M,r) + k)\ (mod\ q)
\end{aligned}$$

$$\begin{aligned}
s_{c_{u-1}} &= (x_{c_1}(A_{c_1} \ldots A_{c_{u-1}}) + \textstyle\sum_{\sigma=2}^{u-1} (x_{c_\sigma} \prod_{j=\sigma-1}^{u-1} A_{c_j}) \\
&\quad + \textstyle\sum_{\sigma=1}^{u-2} (k_{c_\sigma} \prod_{j=\sigma+1}^{u-1} A_{c_j}) + k_{c_{u-1}})\ (mod\ q)
\end{aligned}$$

$$\begin{aligned}
s_{d_{m-1}} &= (x_{d_1}(A_{d_1} \ldots A_{d_{m-1}}) + \textstyle\sum_{\sigma=2}^{m-1} (x_{d_\sigma} \prod_{j=\sigma-1}^{m-1} A_{d_j}) \\
&\quad + \textstyle\sum_{\sigma=1}^{m-2} (k_{d_\sigma} \prod_{j=\sigma+1}^{m-1} A_{d_j}) + k_{c_{m-1}})\ (mod\ q)
\end{aligned}$$

Then, by some calculation, we can express $s_E(M)$ as follows.

$$\begin{aligned}
s_E(M) &= s_C(M) + (s_{d_{m-1}} + x_{c_u} A_{d_{m-1}} - (s_{c_{u-1}} + x_{c_u} A_{c_{u-1)}}))h(M,r) \\
&\quad \textstyle\prod_{i=u}^{n-1} A_{c_i}\ (mod\ q) \\
&= ((x_{d_1}(A_{d_1} \ldots A_{d_{m-1}} A_{c_u} \ldots A_{c_{n-1}}) + \textstyle\sum_{\sigma=2}^{m-1} (x_{d_\sigma} \prod_{j=\sigma-1}^{m-1} A_{d_j} \\
&\quad \textstyle\prod_{j=u}^{n-1} A_{c_j}) + x_{c_u}(A_{c_{u-1}} \ldots A_{c_{n-1}}) + \sum_{\sigma=u+1}^{n-1} (x_{c_\sigma} \prod_{j=\sigma-1}^{n-1} A_{c_j}) \\
&\quad + x_{c_n} A_{c_{n-1}} + \textstyle\sum_{\sigma=1}^{m-2} (k_{d_\sigma} \prod_{j=\sigma+1}^{m-1} A_{d_j} \prod_{j=u}^{n-1} A_{c_j}) \\
&\quad + \textstyle\sum_{\sigma=u-1}^{n-2} (k_{c_\sigma} \prod_{j=\sigma+1}^{n} A_{c_{\sigma+1}}) + k_{c_{n-1}}) h(M,r) + k)\ (mod\ q)
\end{aligned}$$

And one can construct the corresponding proxy public key for user c_n with respect to the delegation chain E easily as follows:
$y_{d_1}{}^{A_{d_1} \ldots A_{d_{m-1}} A_{c_u} \ldots A_{c_{n-1}}} \prod_{\sigma=2}^{m-1} y_{d_\sigma}{}^{A_{d_{\sigma-1}} \ldots A_{d_{m-1}} A_{c_u} \ldots A_{c_{n-1}}} \prod_{\sigma=u}^{n-1} y_{c_\sigma}{}^{A_{c_{\sigma-1}} \ldots A_{c_{n-1}}}$
$\prod_{\sigma=1}^{m-2} r_{d_\sigma}{}^{A_{d_{\sigma+1}} \ldots A_{d_{m-1}} A_{c_u} \ldots A_{c_{n-1}}} \prod_{\sigma=u}^{n-2} r_{c_\sigma}{}^{A_{c_{\sigma+1}} \ldots A_{c_{n-1}}} r_{c_{n-1}}\ (mod\ p)$. ■.

The reason why the attack is successful is that $x_{c_{u+1}}$ is only linked with w_{c_u} and r_{c_u}. However, there is no linkage with the earlier part of the delegation chain. As a result, user u may substitute w_{c_i} and r_{c_i} (where $i < u$) with w_{d_i} and r_{d_i} (where $i < m$) without the awareness of user c_n. Therefore, if user c_n has performed signing of a message M on behalf of user c_1, he/she may also unknowingly perform proxy signing on behalf of another user d_1 on that message. As a result, given a proxy signature $s_C(M)$, it is possible for the proxy signer c_n to deny that he/she signs on behalf of c_1 at a later time. Therefore, strong non-repudiation cannot be supported.

Note that by launching the described attack, the private key of user c_u, x_{c_u}, may be computed from $S_C(M)$, $S_E(M)$, $s_{c_{u-1}}$ and $s_{d_{m-1}}$ (which are considered to be public in the proposed scheme) (See Theorem 2).

Theorem 2 *By launching the attack, the private key of user c_u may be computed as follows.*

$$\begin{aligned}
x_{c_u} &= (s_E(M) - s_C(M) + \textstyle\prod_{i=u}^{n-1} A_{c_i}(s_{c_{u-1}} - s_{d_{m-1}}))/((A_{d_{j-1}} - A_{c_{u-1}}) \\
&\quad h(M,r) \textstyle\prod_{i=u}^{n-1} A_{c_i})\ (mod\ q)
\end{aligned}$$

Proof

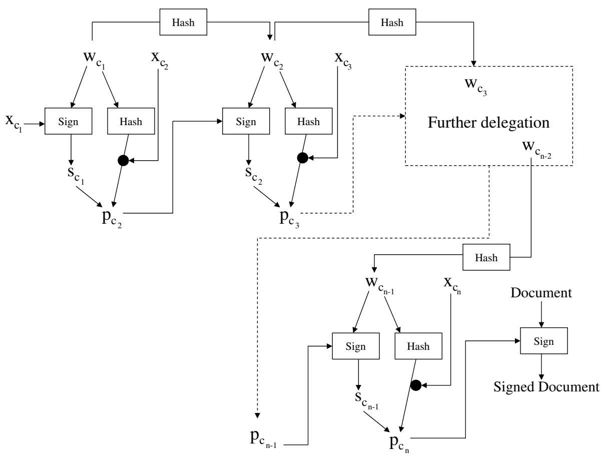


Figure 3. Chained delegation of signing rights (Improved Scheme)

$$\begin{aligned}
R.H.S. &= (s_E(M) - s_C(M) + \textstyle\prod_{i=u}^{n-1} A_{ci}(s_{c_{u-1}} - s_{d_{m-1}}))/((A_{d_{j-1}} - A_{c_{u-1}}) \\
&\quad h(M,r)\textstyle\prod_{i=u}^{n-1} A_{c_i})\ (mod\ q) \\
&= (s_C(M) + (s_{d_{j-1}} + x_{c_u}A_{d_{j-1}} - s_{c_{u-1}} - x_{c_u}A_{c_{u-1)}})h(M,r)\textstyle\prod_{i=u}^{n-1} A_{c_i} \\
&\quad -s_C(M) + \textstyle\prod_{i=u}^{n-1} A_{c_i}(s_{c_{u-1}} - s_{d_{m-1}}))/((A_{d_{j-1}} - A_{c_{u-1}})h(M,r) \\
&\quad \textstyle\prod_{i=u}^{n-1} A_{c_i})\ (mod\ q) \quad \text{(Theorem 1)} \\
&= (((x_{c_u}A_{d_{j-1}} - x_{c_u}A_{c_{u-1}})h(M,r)\textstyle\prod_{i=u}^{n-1} A_{c_i}))/((A_{d_{j-1}} - A_{c_{u-1}}) \\
&\quad h(M,r)\textstyle\prod_{i=u}^{n-1} A_{c_i})\ (mod\ q) \\
&= x_{c_u}\ (mod\ q) \\
&= L.H.S
\end{aligned}$$

However, there are situations when the cost to the attacker is minimal (e.g. consider the scenario where the private key of the attacker will expire very soon but it may be advantageous by convincing a third party that the forged signature is valid before the expiry time). Therefore, the attack outlined in this section should be addressed to protect the proxy signer.

3.3. The improved scheme

The improved delegation scheme is shown in Figure 3. It is based on the scheme as described in Section 3.1. In addition, for all $1 < i < n - 1$, we require that the delegation warrant w_{c_i} to include the cryptographic hash of $w_{c_{i-1}}$, which will be denoted as $h(w_{c_{i-1}})$. A secure cryptographic hash function [11] has the property that given any value x, it is hard to find a $y \neq x$ such that $h(y) = h(x)$. In this case, w_{c_i} is linked to $w_{c_{i-1}}$, which in turn links to all the delegation warrant earlier in the chain in a similar manner. When performing verification of the proxy signature, the verifier should check that for each delegation warrant w_{c_i} where $1 < i < n$, the hash of $w_{c_{i-1}}$ is included.

Consider the case where user c_n performs proxy signing. With this modification to the proxy generation procedure, the attack will invalidate the proxy signature because $w_{c_{n-1}}$ contains $h(w_{c_{n-2}})$, which in turn links to the cryptographic hash of all the previous delegation warrant by the use of hash chain. If the delegation chain C is manipulated in a way that it is substituted by the chain E, the hash chain will be broken. It will be infeasible for the attacker to find another delegation chain E such that the hash chain can be verified correctly if the underlying cryptographic hash function is secure.

3.4. An analysis of the proposed delegation scheme

In this section, we discuss the strong non-repudiation property of the proposed delegation scheme. Consider a delegation chain $C =< c_1, c_2, ..., c_n >$ and a proxy signature signed by the proxy signer (user c_n) with the delegated permission of C. Suppose the length of C is two (i.e. $n = 2$). This is the case which is handled by the basic proxy signature scheme [6]. Because of the strong unforgeability of the underlying proxy signature scheme, the proxy signer cannot falsely deny that he/she generates the signature. Also, by the security of the proxy signature scheme [2], it is not possible for the attacker to convert a normal signature into a proxy signature, convert a proxy signature into a normal signature or replace a proxy associated with a proxy signature with another proxy. Therefore, the proxy signer cannot falsely deny that he/she signs on behalf of the original delegator (user c_1). In addition, because of the verifiability of the basic proxy signature scheme, the original delegator cannot deny that he/she authorizes the signing by performing delegation.

We next discuss the case where $n > 2$. Before our discussion, we enumerate the security properties of the proposed delegation scheme as follows.

- We first discuss whether it is possible for the attacker to convert a normal signature into a valid proxy. Suppose a user generates a signature on a normal message. The attacker can use it as a proxy (signed delegation warrant) to generate a proxy signature or re-delegate. Therefore, a user should never sign on any delegation warrant except for the purpose of performing delegation/re-delegation. To address this potential attack, organizations can impose a different message structure to distinguish between a normal message and delegation warrant to distinguish the intention of the signer. Such measure can also prevent the attacker from converting a proxy (a signature on a delegation warrant) to signature on a normal message.
- A proxy signature for a delegation chain $C =< c_1, c_2, ..., c_n >$ is generated by signing a document with the proxy private key p_{c_n} (which is computed with the proxy $s_{c_{n-1}}$ and the private key of user c_n). By the security [5] of the underlying proxy signature scheme [6], the attacker is not able to remove the proxy $s_{c_{n-1}}$ to form a normal signature or replace it with a proxy for another delegation chain. Also, since the delegation warrants w_{c_i} ($2 < i < n - 1$) are linked with a hash chain, the attacker will not be able to convert the proxy $s_{c_{n-1}}$ for a delegation chain to a proxy for another delegation chain.
- We analyze whether the requirements R1 to R5 in [7] are satisfied for the proposed delegation scheme. Consider a delegation chain $C =< c_1, c_2, ..., c_n >$. We first consider the requirement R1 (Verifiability), which states that from the proxy signature a verifier can be convinced of the original delegator's agreement on the signed message. We extend the original definition of verifiability, which can only handle delegation chain with length equals to two, to handle delegation chains of arbitrary length. Consider the delegation chain C with length n. We require the agreement of all the users $c_1, c_2, ..., c_{n-1}$ to be present in the proxy signature on

M. The proposed delegation scheme satisfies this requirement because the delegation warrant w_{c_i} (for all $1 < i < n-1$) is included in the proxy public key t_n, which is used to verify the proxy signature. If M conforms to all the delegation warrants, the proxy signature is valid.

We next consider the requirement R2 (Strong Unforgeability), which states that only the delegate can create a valid proxy signature on behalf of the original delegator. However, the original delegator and other third parties who are not designated as the proxy signer cannot create a valid proxy signature. In our proposed scheme, in order to generate the proxy private key which is used to perform proxy signing, the private key of the proxy signer, x_{c_n}, should be used. Thus, only the legitimate signer can create a valid proxy signature. In addition, we consider an attack by the original delegator, which is similar to [12]. The goal of this attack is for the original delegator to forge signature of the proxy signer. In the context of the delegation scheme proposed in this paper, suppose user c_1 intends to forge a proxy signature created by user c_n in the delegation chain $C =< c_1, c_2, ..., c_n >$. For each $1 \leq i < n$, user c_1 should choose r_{c_i} such that $r_{c_i} = (y_{c_{i+1}} h(w_{c_i}, r_{c_i}))^{-1} \ (mod\, p)$. However, it is infeasible for the attacker to solve this equation because of the difficulty of inverting the hash function.

Requirement R3 (Strong identifiability), which states that any third party should be able to determine the identity of the proxy signer from a proxy signature. This requirement is also satisfied because the public key of the user c_n is explicitly included in $w_{c_{n-1}}$. Thus, the identity of the proxy signer can be identified from the proxy public key t_{c_n}, which is used to verify the proxy signature.

Requirement R4 (Strong undeniability), which states that once a proxy signer creates a valid proxy signature, he/she cannot repudiate the signature creation. This requirement is also satisfied because the proxy signer user c_n cannot repudiate as the proxy private key p_{c_n}, which is used to generate the proxy signature, can only be generated by himself/herself.

Requirement R5 (Prevention of misuse), which states that the proxy signer cannot use the proxy key for purposes other than generating a valid proxy signature. That is, he/she cannot sign, with the proxy private key, messages that have not been authorized by the original delegator. This requirement is also satisfied because we require the delegation warrant (e.g. the delegator, the delegate and restrictions on how the proxy should be used) to be explicitly specified in the delegation warrant w_{c_i} for all $1 \leq i < n$. Therefore, illegal proxy transfer is prevented and the signing capability of the proxy signer is limited by the delegation warrant.

We now continue with our discussion on the strong non-repudiation property. Because of the unforgeability of the proposed delegation scheme, the proxy signer cannot falsely deny that he/she generates the signature. In addition, the delegation chain $C' =< c_1, c_2, ...c_{n-1} >$ is linked with a cryptographic hash function and by the security of the basic proxy signature scheme [2], the delegation warrant $w_{c_{n-1}}$ is securely linked to the proxy signature. Since the attacker can neither convert a proxy signature into a normal signature nor convert a proxy signature for a delegation chain to another delegation chain, the proxy signer (user c_n) cannot falsely deny that he/she signs on behalf of the original delegator of C (user c_1) by using the delegation permission associated with C. Also, because of the verifiability of the proposed delegation scheme, the proxy signature itself contains the commitment of both the original and intermediate delegators.

The delegators cannot deny that they authorize the signing by performing delegation/re-delegation.

Therefore, the improved scheme supports strong non-repudiation in chained delegation of signing rights. As a remark, there may be other ways to support strong non-repudiation. For instance, the information about the delegation chain can be included in the document to be signed. In this way, the attacker will not be able to manipulate the signature such that the signer will unknowingly perform proxy signing on behalf of another unintended user. However, our proposed scheme does not require the structure of the signed document to be modified to include the delegation chain information. As a result, the semantic of the document to be signed can be preserved.

3.5. Comparison with Delegation Using Hierarchical Tokens [3]

After describing the details of our scheme, we now compare our scheme with the delegation scheme as proposed by Ding and Petersen [3]. The paper classifies delegation schemes into six main classes and propose delegation schemes to address each type of delegation. However, these schemes are similar and only differ in minor details. Among the six schemes, the identity-based traceable delegation is the most common and useful type of delegation in organizations. Therefore, we will focus on analyzing the difference between this scheme with our proposed scheme.

The first difference concerns how the key pairs for users are generated. In the scheme by Ding and Petersen, the private key of the user is generated by the certificate authority (CA). Since the CA is able to forge the signature of the users, it is very difficult to achieve non-repudiation. In contrast, in our proposed delegation scheme, each user in the organization may generate his/her own private key to support non-repudiation.

The second, and the most important, difference concerns how the proxy signature is generated. The scheme proposed by Ding and Petersen does not directly support the creation of proxy signature. A signature created by the proxy signer in this scheme is not distinguishable from a normal signature. In this scheme, the proxy signer uses his/her own private key to sign a document. Although the proxy signer cannot repudiate that he/she signs the document, he/she can repudiate at a later time that he/she signs on behalf of the original delegator and strong non-repudiation is not supported. In contrast, as discussed in the previous section, strong non-repudiation can be supported by our proposed delegation scheme.

Also, our proposed scheme is more efficient than the hierarchical delegation scheme as proposed by Ding and Petersen. For the sake of discussion, we will compare the performance of the two schemes in terms of the number of exponentiation operations (as it is the most computationally expensive operation). For performing delegation and signing, both of the schemes only require multiplication and addition and so their performance is similar. However, our scheme is more efficient in proxy signature verification. For signature verification for a delegation of chain of length n, our scheme require $n-1$ exponentiation for the computation of the proxy public key and 2 additional exponentiation for the verification of the actual proxy signature of the message (i.e. a total of $n+1$ exponentiation operations). However, for the scheme proposed by Ding and Peterson, $1+2(n-1) = 2n-1$ exponentiation operations for the verification of the delegation token and 2 additional exponentiation operations for the verification of signature for the actual message are required (for the fairness of comparison, we ignore the cost for the computation of each user's public key from the public key of the CA) (i.e. a total of $2n + 1$ exponentiation operations). Therefore, our scheme is more efficient than delegation using hierarchical tokens in terms of proxy signature verification.

4. Summary

In this paper, we extend a proxy signature scheme proposed by Kim et al. [6] to support the chained delegation if signing rights. We show that a straight-forward extension of the Kim et al.'s scheme is vulnerable to a new form of attack from an insider, which is detailed in Section 3.2. Because of the attack, strong non-repudiation cannot be supported. We propose a way to improve the scheme to make it resistant to the attack. The proposed delegation scheme supports efficient verification of authorization. It also supports strong non-repudiation without the need for modifying the structure of the signed document.

Acknowledgement

This research is supported in part by the Areas of Excellence Scheme established under the University Grants Committee of the Hong Kong Special Administrative Region, China (Project No. AoE/E-01/99), two grants from the Research Grants Council of the HKSAR, China (Project No. HKU/7144/03E and HKU/7136/04E), and two grants from the Innovation and Technology Commission of the HKSAR, China (Project No. ITS/170/01 and UIM/145).

References

[1] Tuomas Aura. Distributed access-rights management with delegation certificates. In *Secure Internet Programming*, pages 211–235, 1999.

[2] Alexandra Boldyreva, Adriana Palacio, and Bogdan Warinschi. Secure proxy signature schemes for delegation of signing rights. *http://eprint.iacr.org/curr/*, 2003.

[3] Y. Ding, P. Horster, and H. Petersen. A new approach for delegation using hierarchical delegation tokens. In *Proc. 2nd Conference on Computer and Multimedia Security, Chapman and Hall, pp. 128–143*, 1996.

[4] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Crypto '84, LNCS 196, pp. 1018*, 1984.

[5] Seungjoo Kim Jung-Yeun Lee, Jung Hee Cheon. An analysis of proxy signatures: Is a secure channel necessary? In *CT-RSA'03, LNCS 2612, pp. 68-79, 2003. Berlin: Springer-Verlag*, 2003.

[6] S. Kim, S. Park, and D. Won. Proxy signatures, revisited. In *Information and Communications Security (ICICS'97), LNCS 1334, pp. 223-232, 1997. Berlin: Springer-Verlag*, 1997.

[7] B. Lee, H. Kim, and K. Kim. Strong proxy signature and its applications. In *Proc. of SCIS, pp.603-608*, 2001.

[8] Masahiro Mambo, Keisuke Usuda, and Eiji Okamoto. Proxy signatures: Delegation of the power to sign messages. In *IEICE Trans. on Fundamentals, Vol.E79-A, No.9, pp.1338-1354*, 1996.

[9] M.Mambo, K.Usuda, and E.Okamoto. Proxy signatures for delegating signing operations. In *3rd ACM Conf. on Computer and Communication Security, pp. 48–57*, 1996.

[10] C.P. Schnorr. Efficient identification and signatures for smart cards. *Advances in Cryptology – CRYPTO '89, Lecture Notes in Computer Science, Vol. 435, pp. 239-252, Berlin: SpringerVerlag*, 1990.

[11] William Stallings. Cryptography and network security: Principles and practice (3rd edition). *Published by Prentice Hall; 3rd edition*, August 27, 2002.

[12] H. Sun and B. Hsieh. On the security of some proxy signature schemes, 2003. Available at http://eprint.iacr.org/2003/068.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005
© 2005 The authors. All rights reserved.

No Author-Based Selective Receipt in Certified Email with Tight Trust Requirements

Nicolás González-Deleito [1]
Département d'Informatique, Université Libre de Bruxelles, Belgium

Abstract. Kremer and Markowitch introduced in [10] a new property for certified email protocols called *no author-based selective receipt*, and proposed two new protocols respecting it. In this paper we show that these protocols implicitly require the sender of the email to trust the trusted third party to assure this property. We propose a new protocol in which this trusted third party has only to be trusted to assure fairness and timeliness, like in most other exchange protocols. Unfortunately, unlike [10], our protocol does not guarantee the delivery of a non-repudiation of origin evidence to the recipient. We prove that this is impossible to achieve without a fully trusted third party.

Keywords. Certified email, non-repudiation, trust requirements, security protocols

1. Introduction

Designing certified email protocols is one of the most important problems related to the fair exchange of electronic information. Such protocols have to assure that either a message originating from a sender is exchanged for a receipt attesting that the recipient of this message received it, or that neither entity receives any valuable information. If they achieve this goal, they are said to be *fair*. Moreover, a related dispute resolution protocol allows an adjudicator to examine the receipt and, if it is valid with respect to the dispute resolution policy of the certified email protocol, be convinced that the recipient received the email.

First solutions to this problem consisted in the gradual release of information [5] between the sender and the recipient. However, these solutions required to both entities similar computational capacities and implied an intensive network usage. Probabilistic solutions [12] removed the requirement of equivalent computational power, but still consisted in an important number of communication rounds. Deterministic solutions with a constant network usage are obtained when using a trusted third party (TTP) guaranteeing that the exchange will be fair. Depending on its level of involvement, a TTP can be classified as *inline*, *online* and *offline*. Inline and online TTPs are involved in each instance

[1] Correspondence to: Nicolás González-Deleito, Département d'Informatique, Université Libre de Bruxelles, Bd. du Triomphe – CP212, 1050 Bruxelles, Belgium. Tel.: +32 2 650 5589; Fax: +32 2 650 5609; E-mail: ngonzale@ulb.ac.be.

of a protocol. The former ones act as an intermediary between the sender and the recipient, while the latter ones are only needed in selected steps of the protocol. If the TTP is only involved when a problem occurs (for example a network failure or a dishonest entity refusing to respect the protocol), then it is said to be offline. Protocols with such a TTP are often called *optimistic*.

Several papers have addressed the certified email problem either with an inline [2, 15], an online [4,14] or an offline TTP [13,10,16]. A hybrid solution with an online TTP having only to be trusted by the sender and an offline TTP having to be trusted by both participants has also been proposed [1]. Moreover, we note that multi-party extensions with a single sender and a set of recipients have also been considered in the online [9] as in the offline [11,16] settings.

Among all these solutions, the one proposed by Kremer and Markowitch [10] deserves special attention as it explicitly takes into account the anonymity property offered by traditional postal services. More precisely, when we receive a certified mail, the postman asks us to sign the receipt and, once it is signed, he delivers us the mail. But at no moment he informs us about the identity of the sender. Kremer and Markowitch consider that knowing this identity could give the recipient enough information about the content of the mail and could allow him to refuse to sign the receipt, breaking therefore the fairness property. They illustrate this fact by describing the scenario where a person not paying the rent of his flat receives a certified mail from his landlord. If the identity of the sender is known to this person before he signs the receipt, then he would not need to read the message to guess that the landlord is claiming his rent. By refusing to sign the receipt, this person prevents the landlord from obtaining a proof for his claim.

In order to prevent a recipient from deciding whether to receive or not a certified email on the basis of the sender's identity, Kremer and Markowitch introduced a new property called *no author-based selective receipt*. This property is achieved by using anonymous communication channels preserving the identity of the sender from the intended recipient as well as from any other entity listening on the communication channel. Furthermore, these channels should offer the recipient the ability to reply to a received message while not revealing the sender's identity. MIX-nets [3] and Onion routing [7] can, for example, be used in order to obtain such kind of channels. For a more detailed discussion on anonymous communication channels we refer the reader to the original paper.

1.1. Contribution

To the best of our knowledge, apart from [10] only [8] considers the no author-based selective receipt property. Unfortunately, the latter protocol does not respect the timeliness property (which assures that each honest entity is able to fairly end the protocol in a finite amount of time) and cannot thus be used in a practical context. Moreover, it requires an online TTP which has to be contacted by an adjudicator at every dispute resolution.

In this paper we study the no author-based selective receipt property [10] and point out that it removes a trust relationship between a sender and the corresponding recipient, as in a protocol not respecting it the former entity has to trust the recipient not to abandon the protocol on the basis of the sender's identity. We show that the protocols proposed by Kremer and Markowitch remove this trust relationship but create a similar one between the sender and the TTP, as the latter has to be trusted by the former not to reveal his identity to the recipient once the protocol has started. Note that the same trust relationship exists in traditional certified mail, as a sender has to trust the postman not to reveal to the recipient his identity (which is normally written on the envelope) unless the latter signs the receipt.

We propose a new protocol in which none of these two trust relationships exist, and therefore not requiring a fully trusted TTP. Like in most other exchange protocols, in ours the TTP has only to be trusted in order to assure the fairness and timeliness properties. Unfortunately, unlike [10], our protocol does not guarantee the delivery of a non-repudiation of origin evidence to the recipient, i.e. an evidence that the sender is the author of the received message. We informally prove that this is impossible to assure without a fully trusted TTP. This limitation implies a clearer difference between certified email and non-repudiation protocols (in which such an evidence is mandatory).

The remaining of this paper is organised as follows. In the following section we define the different properties related to certified email protocols and introduce the notations that will be used. Section 3 describes the main protocol proposed by Kremer and Markowitch [10] and points out the trust relationship behind the no author-based selective receipt property. In section 4 we describe and analyse our new protocol. Finally, in section 5 we discuss about the absence of a non-repudiation of origin evidence and its implications.

2. Definitions and Notations

Before defining the different properties that a certified email protocol has to respect, let us first define the two main types of evidences that are commonly exchanged during a successful protocol run. Through the paper we will call Alice the sender of a certified email, and Bob the intended recipient.

Definition 1 A *non-repudiation of receipt evidence* is an information intended for Alice, that, when presented to an adjudicator at the beginning of the dispute resolution protocol associated with the certified email protocol during which this evidence was generated, will convince him that Bob has received the email. □

Definition 2 A *non-repudiation of origin evidence* is an information intended for Bob, that, when presented to an adjudicator at the beginning of the dispute resolution protocol associated with the certified email protocol during which this evidence was generated, will convince him that Alice is the author of the email. □

The two following properties must be respected by any certified email protocol.

Definition 3 A certified email protocol is said to be *fair* if and only if at the end of a protocol execution either (1) Alice has received her expected non-repudiation of receipt evidence and Bob has received the email from Alice (as well as the possible corresponding non-repudiation of origin evidence), or (2) Alice has not received her non-repudiation of receipt evidence and Bob has not received any information regarding the email (nor the possible non-repudiation of origin evidence). □

Zhou and Gollmann pointed out [15] that traditional postal services do not provide a non-repudiation of origin evidence. This service is not thus required in a certified email context.

Definition 4 A certified email protocol respects the *timeliness* property if and only if each honest participant is always able to reach, in a finite amount of time, a point in the protocol where he can stop the protocol without loosing fairness. □

The following properties are optional.

Definition 5 A certified email protocol provides the *data confidentiality* property if and only if Alice and Bob are the only entities that can obtain the content of the certified email from the messages exchanged during the protocol. □

Definition 6 [10] A certified email protocol achieves the *no author-based selective receipt* property if and only if once the identity of Alice is known to Bob, Bob cannot prevent the delivery of a non-repudiation of receipt evidence to Alice. □

Through the remaining of this paper we will use the following notations:

- $X \rightarrow Y: \quad i$ denotes entity X sending information i to entity Y;
- $X \mapsto Y: \quad i$ denotes X sending information i to Y through a communication channel providing anonymity for X;
- $X \leftarrow\!\!\mid Y: \quad i$ denotes Y replying to X through an already opened communication channel providing anonymity for X;
- $h(i)$ is the result of applying i to a one-way collision-resistant hash function h;
- $E_k(p)$ and $D_k(c)$ are the result of applying respectively a symmetric encryption algorithm E to the plaintext p under the secret key k and a symmetric decryption algorithm D to the ciphertext c under the secret key k;
- $E_X(p)$ is the result of applying an asymmetric (deterministic) encryption algorithm E to the plaintext p under X's public key;
- $S_X(i)$ denotes the digital signature of X over information i (in the description of a protocol's message, $S_X(\star)$ denotes the digital signature of X over all information preceding this signature);
- f_x is a publicly known flag indicating the purpose of a message in a given protocol, where x identifies the corresponding message in that protocol;
- $label$ is an information identifying a protocol run (it will be precisely defined for each protocol);
- m is the electronic mail that Alice sends to Bob.

3. Kremer and Markowitch's Protocol

In this section we briefly describe the optimistic certified email protocol proposed by Kremer and Markowitch [10]. This protocol respects the defined above no author-based selective receipt property, but, as we will show, at the price of additionally asking Alice to trust the TTP not to reveal her identity to Bob.

The protocol supposes that the communication channel used between Alice and Bob is unreliable, while those used between the TTP and, respectively, Alice and Bob, are resilient (i.e. messages sent are correctly received within a finite but unknown amount of time). The value of $label$ is set to $h(m, A, B, k)$.

3.1. Main Protocol

The main protocol begins by Alice sending to Bob through a communication channel providing sender's anonymity a commitment to the mail m to be sent. This first message consists of the identity of the TTP to be contacted in case of problems, the label identifying the protocol run, the ciphered value $c = E_k(m)$ of m under a symmetric key k chosen by Alice and the evidence of origin EOO of m ciphered under the TTP's public key, $E_{TTP}(f_{\mathsf{EOO}}, A, B, label, k, \mathsf{EOO})$, where $\mathsf{EOO} = S_A(f_{\mathsf{EOO}}, A, B, TTP, label, h(c), k)$.

Upon receipt of this first message, if Bob agrees to receive the certified electronic mail from Alice, he replies to her message, through the opened anonymous communication channel, by sending the evidence of receipt for the ciphertext c, $\mathsf{EOR}_c = S_B(f_{\mathsf{EOR}_c}, B, TTP, label, h(c), E_{TTP}(f_{\mathsf{EOO}}, A, B, label, k, \mathsf{EOO}))$.

If Alice receives a valid message from Bob at the second step of this protocol, then she sends to him, through a conventional communication channel, the key k and the evidence of origin EOO. Finally, Bob acknowledges to Alice the proper receipt of the third message by sending an evidence of receipt for k, $\mathsf{EOR}_k = S_B(f_{\mathsf{EOR}_k}, A, B, label, k)$.

1. $A \mapsto B:$ $f_{m_1}, B, TTP, label, c, E_{TTP}(f_{\mathsf{EOO}}, A, B, label, k, \mathsf{EOO})$
2. $A \leftarrow\!\!\!\dashv B:$ $f_{\mathsf{EOR}_c}, label, \mathsf{EOR}_c$
3. $A \rightarrow B:$ $f_{\mathsf{EOO}}, A, B, label, k, \mathsf{EOO}$
4. $B \rightarrow A:$ $f_{\mathsf{EOR}_k}, A, B, label, \mathsf{EOR}_k$

3.2. Recovery Protocol

The recovery protocol can be invoked either by Bob if he does not receive the third message of the main protocol or by Alice if she does not receive the fourth message of this protocol (in the protocol summary below, X denotes the entity invoking the protocol). In both cases, the initiator of this recovery protocol has to send to the TTP the information that has been exchanged during the first two steps of the main protocol, that is, the label, the hash value of c, the evidence of receipt for c and the evidence of origin of m ciphered under the TTP's public key.

The TTP only considers requests for protocol runs having not already been recovered or aborted. As the TTP cannot verify the label (because it cannot recover m from $h(c)$), it further identifies a protocol run by appending to the label the identities of Alice and Bob (in the order found in the ciphered evidence of origin). Therefore, if the protocol run can be recovered, the TTP verifies the signatures corresponding to EOO and EOR_c. If they are both valid, it sends to Alice the evidence of receipt for c and a confirmation of receipt for k, $\mathsf{Con}_k = S_{TTP}(f_{\mathsf{Con}_k}, A, B, label, k)$, replacing the corresponding evidence of receipt having to be issued by Bob in a faultless scenario. Moreover, the TTP sends to Bob the key k and the evidence of origin of m.

1. $X \rightarrow TTP:$ $f_{r_1}, B, label, h(c), \mathsf{EOR}_c, E_{TTP}(f_{\mathsf{EOO}}, A, B, label, k, \mathsf{EOO})$
2. $TTP \rightarrow A:$ $f_{r_A}, A, B, label, k, \mathsf{Con}_k, \mathsf{EOR}_c$
   $TTP \rightarrow B:$ $f_{r_B}, A, B, label, k, \mathsf{EOO}$

3.3. Abort Protocol

The abort protocol can only be invoked by Alice if ever she does not receive the second message of the main protocol. If so, she has to send to the TTP, through a communication channel providing sender's anonymity, a signed abort request enciphered under the TTP's public key, containing the label and the identities of Alice and Bob. The use of encryption and of an anonymous communication channel between Alice and the TTP prevents Bob from tracing abort requests arriving to the TTP back to their sender.

Again, the TTP only considers abort requests if the corresponding protocol run (identified by the label and the identities of Alice and Bob, in this order) has not been recovered or aborted. Thus, if the protocol run can be aborted, then the TTP verifies the signature contained in the abort request. And if it is valid, it sends a signed abort confirmation to both Alice and Bob.

1. $A \mapsto TTP:$ $E_{TTP}(f_{a_1}, A, B, label, S_A(f_{a_1}, label))$
2. $A \leftarrow\!\!\!\shortmid\; TTP:$ $f_{a_2}, label, S_{TTP}(\star)$
   $TTP \rightarrow B:$ $f_{a_2}, label, S_{TTP}(\star)$

3.4. Dispute Resolution Protocols

The non-repudiation of origin evidence is only composed of EOO. However, the content of the non-repudiation of receipt evidence depends on whether the TTP has taken part in the exchange or not. This second evidence is equal to either $(\mathsf{EOR}_c, \mathsf{EOR}_k)$ or $(\mathsf{EOR}_c, \mathsf{Con}_k)$. We refer the reader to the original paper [10] for the details about the dispute resolution protocols.

3.5. About the No Author-Based Selective Receipt Property

As shown by Kremer and Markowitch [10], the above protocol respects the no author-based selective receipt property. The authors also proposed a variant protocol providing moreover the data confidentiality property. But both protocols respect the former property under the hypothesis that the TTP used in case of problems is fully trusted. If we consider a more reasonable and realistic scenario where the TTP is only trusted to assure the fairness and timeliness properties (like in most other exchange protocols), then this protocol may fail to provide the no author-based selective receipt property.

Let us suppose that the TTP used in order to successfully realise the exchange may misbehave but that it will always assure fairness. Then Bob could conspire with the TTP in order to learn the identity of the sender of the certified email before even taking part in the protocol. The attack could take place as follows. Alice performs the first step of the main protocol, but Bob does not reply to it, forcing therefore Alice to run the abort protocol. In this protocol, Alice sends a signed abort request in order to allow the TTP to be sure that she is the entity that started the main protocol and the only one that can invoke the abort protocol. The TTP, before going on with the abort protocol, transmits to Bob the signed request received from Alice and waits for a reply. Bob can then decide either to perform the recovery protocol or let the abort protocol execution to be completed, according to the identity of Alice. The no author-based selective receipt property is therefore not respected.

A similar attack could be realised by having Bob asking the TTP to decipher $E_{TTP}(f_{\mathsf{EOO}}, A, B, label, k, \mathsf{EOO})$ and informing him of the identity of Alice (the TTP cannot reply with the whole information obtained after deciphering as this would imply sending k to Bob and break the fairness property, which would contradict our hypothesis).

A more subtle scenario arises when looking at the third step of the main protocol. If Alice decides not to send the corresponding message to Bob and performs instead a recovery, then the TTP could transmit her identity to Bob and either perform the second step of this recovery protocol or send an error message, depending on Bob's reply. Both possibilities would be fair, but, of course, the second one would allow Alice to detect a misbehaviour from the TTP. However, it is important to note that the TTP cannot be sure that Bob has not received his expected EOO. It is thus not prudent for the TTP to take part in this attack, as the fairness property may not be always respected. However, this third attack would not have been possible in a protocol with three steps in a faultless execution [13,16], as the identity of Alice would have only been revealed once she would have received a *complete* non-repudiation of receipt evidence.

Like in the certified mail service proposed by traditional postal services, the Kremer and Markowitch's protocol needs a fully trusted TTP, as Alice has not only to trust the TTP to assure fairness and timeliness, but she has also to trust it to guarantee the no author-based selective receipt property. However, in favour of the Kremer and Markowitch's protocol we can say that it uses an offline TTP, unlike the inline model of traditional postal services.

4. A Protocol with Tight Trust Requirements

We now present a new optimistic protocol in which Alice does no longer need to trust the TTP to guarantee the no author-based selective receipt property. As in the original protocol [10], the communication channel between Alice and Bob is supposed to be unreliable, while those between the TTP and, respectively, Alice and Bob are supposed to be resilient.

However, in order to prevent the attacks described above when using a TTP that is not fully trusted, our protocol is composed of three steps in a faultless scenario and Alice is required not only to contact the TTP by using a communication channel providing sender's anonymity, but also to avoid sending any information allowing the TTP to learn her identity.

Furthermore, it has to be possible for the TTP to discard requests corresponding to an already aborted or recovered protocol run. As it will be explained below, our solution consists in allowing the TTP to (partially) verify the correct construction of the label identifying a protocol run for each incoming request. This leads us to define the label as $label = h(h(r), B, h(A, m), h(E_B(k)))$, where r is a value randomly chosen by Alice at the beginning of a protocol run. Note that all the information identifying a protocol run (the identities of Alice and Bob, the email and the deciphering key) are included in this label. Furthermore, the email and the identity of Alice are included in the label as $h(A, m)$. This enables the partial verification of the label without knowing the values of A and m, and will allow the protocol to respect the no author-based selective receipt property.

4.1. Main Protocol

As in the original protocol, Alice begins this protocol by sending to Bob, through a communication channel providing sender's anonymity, a commitment to m containing the identity of the TTP, the label identifying the protocol run, the ciphered value $c = E_k(m)$ of m under a symmetric key k chosen by Alice and $C = E_{TTP}(B, label, h(r), h(A, m), E_B(k))$. Note that, compared to the original protocol, from the latter value the identity of Alice cannot be derived. In particular, this value does not include an evidence of origin (which is not required in a certified email protocol). This prevents the TTP from obtaining the identity of Alice and forward it to Bob.

If Bob accepts the email sent by Alice, he replies to her message by sending through the opened anonymous communication channel EOR_c, the evidence of receipt for the ciphertext c, which constitutes the non-repudiation of receipt evidence expected by Alice, $\mathsf{EOR}_c = S_B(f_{\mathsf{EOR}_c}, B, TTP, label, h(c), C)$.

Finally, if the digital signature received from Bob is valid with respect to the information sent to him during the first step of the protocol, then Alice replies through a conventional communication channel with $h(r)$ and $E_B(k)$.

1. $A \mapsto B: \quad f_{m_1}, B, TTP, label, c, C$
2. $A \leftarrow\!\!\shortmid B: \quad f_{\mathsf{EOR}_c}, label, \mathsf{EOR}_c$
3. $A \rightarrow B: \quad f_{m_3}, A, B, label, h(r), E_B(k)$

4.2. Recovery Protocol

At the end of a successful execution of the main protocol, Bob first deciphers $E_B(k)$, then deciphers c with the help of k, and finally verifies whether the label is equal to $h(h(r), B, h(A, D_k(c)), h(E_B(k)))$. This test allows him to be sure that the information he has received during the last step of the main protocol is coherent with the information exchanged during the two previous steps of this protocol. If it fails, or if he has not received the third message of the main protocol, Bob will have to run the recovery protocol by sending to the TTP the information needed to verify that the first two steps of the protocol have been performed, that is $h(c)$, C and EOR_c.

The TTP will therefore decipher C in order to obtain $h(r)$, $h(A, m)$ and $E_B(k)$ and verify if the label has been properly constructed, in which case the recovery request can be considered. If this verification is successful, the TTP further tests if the protocol has not been recovered or aborted. If so, the TTP checks whether EOR_c is valid with respect to the entity B contained in C, in which case it stores EOR_c and sends $h(r)$, $h(A,m)$ and $E_B(k)$ to this B. Otherwise, if one of these tests fails, the TTP sends to Bob an error message informing him about the unsuccessful test.

1. $B \rightarrow TTP:\quad f_{r_1}, TTP, label, h(c), C, \mathsf{EOR}_c$
2. if not already recovered or aborted
   $TTP \rightarrow B:\quad f_{r_B}, B, label, h(r), h(A,m), E_B(k)$

4.3. Abort Protocol

The abort protocol has to be invoked by Alice if ever she does not receive a satisfactory response from Bob at the second step of the main protocol. As we have seen, Alice has to send to the TTP an abort request not containing any information allowing the TTP to learn her identity. Furthermore, this request has to be such that it can only be issued by Alice (this was accomplished in the original protocol by a digital signature). We achieve this by having Alice to send to the TTP, through a communication channel providing sender's anonymity, an abort request message containing r, B, $h(A,m)$, $h(E_B(k))$ and $label$. Note that the value of r is revealed during the abort protocol in order to guarantee that Alice is the entity having started the main protocol (if $h(r)$ was sufficient, a dishonest Bob could realise an abort after a successful execution of the main or the recovery protocols). Also note that, contrary to the original protocol, Alice's request does not need to be ciphered under the TTP's public key, as her identity cannot be derived from the information contained in this request.

If $label$ can be properly verified, then the abort request can be considered. If so, the TTP checks whether the protocol has already been recovered, in which case it sends to Alice (through the opened anonymous communication channel) the EOR_c signature stored during the recovery protocol. If the protocol has not been recovered or aborted, the TTP confirms to Alice (anonymously) and to Bob (through a conventional communication channel) that the protocol run identified by $label$ has been aborted, by sending to them r, B, $h(A,m)$, $h(E_B(k))$ and $label$, along with its signature on these information.

Otherwise, if the abort request cannot be considered or if the corresponding protocol run has already been aborted, the TTP replies to Alice with an error message informing her about the unsuccessful test.

1. $A \mapsto TTP:\quad f_{a_1}, r, B, h(A,m), h(E_B(k)), label$
2. a. if recovered
   $A \leftarrowtail TTP:\quad f_{r_A}, label, \mathsf{EOR}_c$
   b. else if not aborted
   $A \leftarrowtail TTP:\quad f_{a_2}, r, B, h(A,m), h(E_B(k)), label, S_{TTP}(\star)$
   $TTP \rightarrow B:\quad f_{a_2}, r, B, h(A,m), h(E_B(k)), label, S_{TTP}(\star)$

4.4. Dispute Resolution Protocol

If Alice claims to have successfully sent an email m to Bob and the latter denies having received it, Alice has to provide to an adjudicator EOR_c, m, k, $h(r)$, her identity and the identities of Bob and the TTP. The adjudicator reconstructs from these information $E_B(k)$, *label*, C and c, and verifies whether EOR_c is a valid Bob's signature on $(f_{\mathsf{EOR}_c}, B, TTP, \mathit{label}, h(c), C)$. If this test fails, then the adjudicator settles that Bob did not receive the email. Otherwise, the adjudicator has to ask Bob if he can provide a valid abort confirmation token, as a dishonest Alice could abort the protocol after having received EOR_c. If so, it concludes that Bob did not receive the email. Otherwise, it concludes that Bob received m.

As this protocol is currently described, a dishonest Bob could successfully masquerade any other Alice during a dispute resolution by constructing the information that would have been exchanged during a real protocol execution. In order to prevent this, the dispute resolution protocol has to require that Alice signs her request. The adjudicator further verifies whether this signature is valid and whether the entity having performed it corresponds to the identity of Alice allowing to successfully verify the label. We note that this signature could thus be considered as a non-repudiation of origin evidence.

4.5. Analysis

We sketch in the following that our protocol respects the fairness, timeliness, no author-based selective receipt and data confidentiality properties.

Fairness After the first step of the main protocol, Bob can safely stop it if he does not accept to receive a certified email from Alice. If so, Alice has to contact the TTP in order to abort the protocol and prevent Bob from recovering it later. The TTP will therefore abort the corresponding protocol run and confirm this to both Alice and Bob. Alice will not receive her expected non-repudiation of receipt evidence and Bob will not receive the email.

Otherwise, if Bob accepts to receive the email, he could invoke the recovery protocol. In order for this protocol to succeed, he has to provide a valid evidence of receipt as well as the information received from Alice. If the protocol can be recovered, the TTP sends to Bob the key allowing him to obtain the expected email and stores the evidence of receipt for Alice, who will receive it when aborting the protocol. Alice and Bob will thus receive their expected items.

Now, if Bob sends the second message of the main protocol, Alice has to check whether the received signature is valid. Independently of the result of this test, Alice could decide to abort the protocol run (note that she has to do so if the verification fails). This will be possible if Bob has not already launched a valid recovery protocol, in which case Alice and Bob will receive an abort confirmation token that will cancel the possible valid evidence of receipt received by Alice. Note that Bob waits, at this moment, for a response in order to safely stop the protocol. He will therefore receive the abort confirmation token from the TTP, and will be able to present it to an adjudicator during a dispute resolution protocol if ever Alice pretends to have sent the corresponding certified email to Bob.

On the other hand, if the signature verification succeeds, Alice could also either stop the protocol or send the third message to Bob. If after a reasonable amount of time he does not receive any message or if he receives a message that is not valid, Bob has to run the recovery protocol as Alice has already received her expected evidence of receipt while he has not still derived the corresponding email. The TTP will thus send to Bob the key allowing him to obtain this email.

Timeliness From the above discussion on fairness, we can see that, thanks to the underlying resilient communication channels, timeliness is also provided as at each moment both entities can stop the protocol without loosing fairness.

No Author-Based Selective Receipt After the first step of the main protocol, Bob is not able to deduce the identity of Alice as it cannot be implied by the received information. Moreover, these information arrived through a communication channel providing sender's anonymity.

Three scenarios are possible from that point. If Bob decides to stop the protocol, Alice will run the abort protocol and both of them will receive an abort confirmation token. The information sent by Alice and being contained in this token allow the verification of the label, but without revealing Alice's identity. In order to prevent Bob and the TTP from tracing incoming requests at the TTP, Alice contacts the latter by using a communication channel providing sender's anonymity. Therefore, any coalition between Bob and the TTP in order to obtain Alice's identity will not succeed.

If after receiving the first message of the main protocol, Bob decides to perform a recovery, the TTP will send him the key allowing him to obtain the expected email. But, unless this email reveals Alice's identity, he will not be able to deduce it from the other information contained in the TTP's reply. Anyway, if the recovery can be performed, Bob would already have sent to the TTP the non-repudiation of receipt evidence expected by Alice. As after deciphering C the TTP cannot derive Alice's identity, a coalition between Bob and the TTP will be useless.

Finally, if Bob performs the second step of the main protocol, either he will receive an abort token from the TTP, he will have to run the recovery protocol or he will receive from Alice the deciphering key. As we have seen, in the two former cases Bob cannot obtain Alice's identity. In the latter one, Bob receives the key allowing him to obtain the email through a conventional communication channel that enables him to known Alice's identity. But, at this moment, Alice has already received her expected non-repudiation of receipt evidence.

We can therefore conclude that the no author-based selective receipt property is respected in the three possible scenarios described above. Moreover, as Alice's identity is not known by the TTP, the protocol does not require Alice to trust the TTP not to reveal her identity to Bob.

Data Confidentiality As the key k under which m is ciphered is always sent during the protocol encrypted under Bob's public key, it is easy to see that the protocol provides the data confidentiality property.

This property is sometimes seen in the literature as too much expensive for non sensitive messages, although it allows to further reduce trust requirements on the TTP. It is important to note that if the above protocol did not provide it (each occurrence of $E_B(k)$ would therefore have been replaced by k), then Bob could ask the TTP to decrypt C and c, and ask it to derive the identity of Alice from the email m (if possible). In order to respect the no author-based selective receipt property, the key k has to be sent by Alice encrypted for Bob.

5. On the Absence of an Evidence of Origin

The protocol presented in the previous section does not guarantee the delivery of an evidence of origin to Bob, as did the original protocol by Kremer and Markowitch. Although such an evidence may be interesting to allow Bob to prove that he has received an email from Alice, it is not required in the framework of certified email protocols. However, it would be a mandatory element for using this protocol in a non-repudiation context. In the following, we informally prove that it is not possible to realise a (deterministic) non-repudiation protocol providing the no author-based selective receipt property while using a TTP having only to be trusted to assure fairness and timeliness.

We do not consider problems related to the underlying communication channels quality, and therefore suppose that all the communication channels are operational, i.e. messages sent are correctly received within a known, finite and constant amount of time. The fairness property for a non-repudiation protocol states that, at the end of a protocol execution, either (1) Alice has received her expected non-repudiation of receipt evidence and Bob has received the expected message from Alice as well as its corresponding non-repudiation of origin evidence, or (2) Alice has not received her non-repudiation of receipt evidence, and Bob has not received any information regarding the message nor the corresponding non-repudiation of origin evidence.

If we want to respect the no author-based selective receipt property without requiring to trust the TTP to assure it, the protocol also needs to guarantee that once the identity of Alice is revealed to the TTP, the latter will not be able to prevent the delivery of the non-repudiation of receipt evidence to Alice. Therefore, in order to achieve this, Alice must not reveal her identity before having received the non-repudiation of receipt evidence she expects. In other words, she must not send a non-repudiation of origin evidence before having received the corresponding non-repudiation of receipt evidence. Otherwise, Bob could learn (possibly through the TTP) her identity and would be able to decide whether to stop the protocol or not, breaking the no author-based selective receipt property.

On the other hand, if Alice gets the non-repudiation of receipt evidence before sending the non-repudiation of origin evidence, she could decide not to send this latter evidence and stop the protocol, breaking therefore the fairness property. However, one could imagine that in such a situation the TTP could issue a token to cancel the evidence received by Alice. But if the TTP is given this power, it could also use it on Bob's request in order to cancel both evidences once Alice has provided her non-repudiation of origin evidence, and before the end of the protocol (as Bob must not obtain the message in this case). The TTP cannot thus be able to cancel any evidence.

We can see that, in order to solve the original problem, we have to realise a fair exchange of a non-repudiation of origin evidence for a non-repudiation of receipt evidence without a TTP. As our goal is to remove an existing trust relationship, we cannot use a second TTP, as this would replace this trust relationship by another one. Note that this new problem does not take into account the message from the original problem. A protocol solving the first problem would have to assure that Bob receives this message once the second problem has been successfully solved.

As it is not possible to perform a (deterministic) fair exchange of electronic information between two potentially dishonest parties without a TTP [6], we conclude that it is not possible to realise a (deterministic) non-repudiation protocol respecting the no author-based selective receipt property without requiring to trust the TTP to assure it.

However, we note that in our protocol the email m of Alice could be composed of a document and of a non-repudiation of origin evidence for this document. Although it is not possible to guarantee that the email contains such an evidence without knowing the sender's identity, an honest Alice could provide this evidence in order to convince Bob that the email comes from her.

Acknowledgements

I am grateful to Olivier Markowitch for the useful discussions and for his invaluable comments on several drafts of this paper. I would also like to thank Olivier Cailloux for his interesting remarks on one of the final drafts.

References

[1] G. Ateniese, B. de Medeiros, and M. T. Goodrich. TRICERT: A Distributed Certified E-Mail Scheme. In *Proceedings of 2001 Network and Distributed System Security Symposium (NDSS 2001)*. Internet Society, Feb. 2001.

[2] A. Bahreman and J. D. Tygar. Certified Electronic Mail. In *Proceedings of 1994 Network and Distributed System Security Symposium (NDSS 1994)*, pages 3–19. Internet Society, Feb. 1994.

[3] D. L. Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–88, Feb. 1981.

[4] R. H. Deng, L. Gong, A. A. Lazar, and W. Wang. Practical Protocols for Certified Electronic Mail. *Journal of Network and Systems Management*, 4(3):279–297, Sept. 1996.

[5] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts (Extended Abstract). In *Proceedings of Advances in Cryptology – Crypto'82*, pages 205–210. Plenum Publishing, 1982.

[6] S. Even and Y. Yacobi. Relations among public key signature systems. Technical Report 175, Technion – Israel Institute of Technology, Mar. 1980.

[7] D. M. Goldschlag, M. G. Reed, and P. F. Syverson. Hiding Routing Information. In *Information Hiding*, volume 1174 of *Lecture Notes in Computer Science*, pages 137–150. Springer-Verlag, May 1996.

[8] K. Imamoto and K. Sakurai. Private Certified E-mail Systems with Electronic Notice Board. In *Proceedings of the 9th International Conference on Distributed Multimedia Systems (DMS 2003)*, pages 726–729. Knowledge Systems Institute, Sept. 2003.

[9] S. Kremer and O. Markowitch. A multi-party non-repudiation protocol. In *Proceedings of the 15th International Conference on Information Security*, Information Security for Global Information Infrastructures, pages 271–280. Kluwer Academic Publishers, Aug. 2000.

[10] S. Kremer and O. Markowitch. Selective Receipt in Certified E-Mail. In *Proceedings of the 2nd International Conference on Cryptology in India (Indocrypt 2001)*, volume 2247 of *Lecture Notes in Computer Science*, pages 136–148. Springer-Verlag, Dec. 2001.

[11] O. Markowitch and S. Kremer. A Multi-Party Optimistic Non-Repudiation Protocol. In *Proceedings of the 3rd International Conference on Information Security and Cryptology*, volume 2015 of *Lecture Notes in Computer Science*, pages 109–122. Springer-Verlag, Dec. 2000.

[12] O. Markowitch and Y. Roggeman. Probabilistic Non-Repudiation without Trusted Third Party. In *Proceedings of the 2nd Conference on Security in Communication Networks (SCN'99)*, Sept. 1999.

[13] S. Micali. Simple and Fast Optimistic Protocols for Fair Electronic Exchange. In *Proceedings of the 22nd annual symposium on Principles of Distributed Computing*, pages 12–19. ACM Press, July 2003.
[14] B. Schneier and J. Riordan. A Certified E-Mail Protocol. In *Proceedings of the 14th Annual Computer Security Applications Conference*. ACM Press, Dec. 1998.
[15] J. Zhou and D. Gollmann. Certified Electronic Mail. In *Proceedings of 1996 European Symposium on Research in Computer Security (ESORICS'96)*, volume 1146 of *Lecture Notes in Computer Science*, pages 160–171. Springer-Verlag, Sept. 1996.
[16] J. Zhou, J. Onieva, and J. Lopez. Optimized Multi-Party Certified Email Protocols. *Information Management & Computer Security*, 2005.

Authorization & Access Control

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

A Generic Protocol for Controlling Access to Mobile Services

Shuhong Wang [a], Yingjiu Li [a,1], Bo Zhu [b], and Nan Hu [a]
[a] *Singapore Management University*
[b] *Institute for Infocomm Research, Singapore*

Abstract. Mobile services have been growing fast to facilitate business in wireless network environment. It is both critical and challenging to maintain security and anonymity so as to provide high quality services. In this paper, we propose a ticket-based architecture and a generic protocol for controlling access to mobile services. Our protocol has the following properties. First, it is a generic solution independent of cryptographic algorithms and service models. Second, it is secure against various malicious attacks on mobile services. Third, it provides identity anonymity for customers and/or service providers depending on business requirements. Fourth, it is flexible in dynamic environments where customers and/or service providers are cross multiple domains. We also show an efficient implementation option of this generic protocol based on elliptic curve digital signature algorithm.

Keywords. Mobile service, security, anonymity, ticket

1. Introduction

The fast development of mobile communication systems — the current implementation of 3G systems, the future deployment of 4G systems, and the inter-connection of a multitude of diverse wireless networks — has greatly facilitated customers to access mobile services and applications at anytime from anywhere. It has been highlighted that the highly-personalized, context-aware, location-sensitive, time-critical applications, conducted in a very secure manner are the most promising mobile services [15]. Such services are technically possible as modern mobile devices such as 3G cell phones and PDAs are usually equipped with advanced functionalities (e.g., browsing the internet) and powerful computing capabilities (which can be used for conducting cryptographic operations such as public key encryptions).

To provide high quality mobile services in the wireless environment, one needs not only convenience and flexibility, but also security and privacy in the design of architectures and protocols. Although there are many proposals for securing mobile services (e.g., [1,4,10,14,16,18,20,22,25–27,34–36]), most of them only provide part of desirable properties. For example, the protocols proposed in [34–36] are vulnerable to forgery and modification attacks (see section 7.1 for details), while [10, 22] do not provide identity anonymity for customers. In addition, many solutions are restricted to specific techniques

[1]Correspondence to: Yingjiu Li, School of Information Systems, Singapore Management University, 80 Stamford Road, Singapore 178902. Tel.: +65 6828 0913; Fax: +65 6828 0919; E-mail: yjli@smu.edu.sg.

or particular types of services. As examples, the security module introduced in [10] is customized for WAP gateway, and the protocols in [34–36] are based on a specific cryptographic algorithm.

In this paper, we propose a ticket-based architecture as well as a generic protocol for controlling access to mobile services. A ticket is a piece of information that allows a customer to access a particular type of services. Our protocol has the following properties. First, it is a generic solution independent of cryptographic algorithms and service modes. Second, it is secure against various malicious attacks on mobile services. Third, it provides identity anonymity for customers and/or service providers depending on business requirements. Fourth, it is flexible in dynamic environments where customers and/or service providers are cross multiple domains. We also show an efficient implementation option of this generic protocol based on elliptic curve digital signature algorithm.

The rest of the paper is organized as follows. In Section 2, we classify attacks on mobile services and define the goals of our work. The overall architecture and generic protocol are presented in Section 3 and Section 4, respectively. Following that in Section 5 is presented an implementation option based on elliptic curve digital signature algorithm. In Section 6, we analyze the properties achieved in our solution. In Section 7, we present the related work and compare them with our solution. Finally, in Section 8, we draw conclusion.

2. System Model

2.1. Types of Attacks

Possible attacks on mobile services can be characterized into the following four types:

- *Type I (use-without-pay)*: Adversaries try to use services without paying for them. Such adversaries could be either outside attackers who are not involved in the transaction or even malicious users within the system.
- *Type II (get-paid-without-serve)*: Service providers try to subvert the system so that they get paid without providing the service.
- *Type III (outsider-to-know-insider)*: Outside attackers launch passive attacks to uncover the real identities of entities involved in the transaction.
- *Type IV (insider-to-know-insider)*: One entity involved in the transaction, either customer, service provider, or trusted credential, intends to get the real identities of other users in the system, although those users wish to be anonymous.

In addition, mobile services may subject to common network attacks such as denial-of -service attacks and replay attacks.

2.2. Design Goals

The goals or properties that are expected in our work are:

- *Generic Protocol*. The protocol for controlling access to mobile services should be compatible with various cryptographic algorithms and service models.
- *Security*. The solution should be secure against Type I attack (use-without-pay) and Type II attack (get-paid-without-serve) as well as common network attacks on mobile services.

- *Anonymity*. There are a few different types of anonymity that should be achieved in mobile services. One is to ensure the identities of two transaction parities (i.e. both the customer and the service provider) anonymous to entities outside the interaction process. Another type of anonymity requires an additional property that even the two interactive parties, i.e. customer and service provider, have no idea about the real identity of each other. Besides that, the identities of two transaction parities are expected to be anonymous to the Trust Credential, who maintains public information in our protocol. The attacks on anonymity can be categorized into Type III attack (outsider-to-know-insider) and Type IV attack (insider-to-know-insider).
- *Scalability*. To offer access to any service at anytime from anywhere, the solution should be able to provide seamless services while the customer roams among different wireless network domains.
- *Efficiency*. With relative limited computation and storage resources, the process on mobile devices, especially at the customer side, should be computationally efficient and require as less storage as possible.

3. The Ticket-based Architecture

There are four roles in our proposed architecture, namely trusted authority (TA), trusted credential (TC), customer (C) and provider (P). We use the name user (U) when we consider it to be either C or P.

In our solution, it is assumed that (1) TA is fully trusted by all the other roles; (2) TC is trusted to undertake the process of charging bills, making clearance, and updating public information correctly. However, it may be curious about the real identities of entities involved in the transactions; (3) Both C and P do not have trusts on each other, and wish to be anonymous to all other entities except TA.

The proposed architecture is shown in Figure 1.

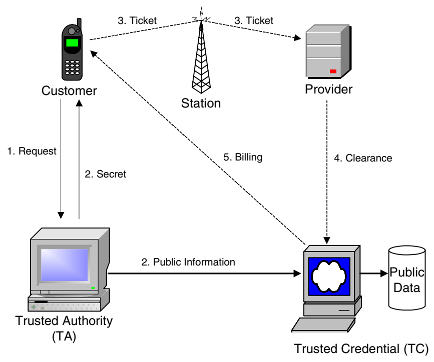


Figure 1. The basic architecture.

Customers and providers need to register with the TA so as to be involved in mobile services. In the registration process, the real identities of all participants are verified. A pair of secret and public information are then generated for each registered entity, along with a virtual identity which is to be used in future transactions for anonymous reason (if the involving entity does not care about anonymity, the virtual identity can be real identity).

When a customer intends to access a provider's services, the former is required to pass over an appropriate ticket by which the latter can check the information about the service being requested. The ticket is generated by the customer itself with its secret key.

After receiving the ticket, the provider checks its validity and provides services according to the ticket. The ticket is then forwarded to TC for charging bill to the customer (payment to TC) and making clearance (payment to the service provider). The details of billing procedure is beyond the scope of this paper, for which many secure electronic payment protocols (e.g., [25, 38]) can be used. In the end, detailed information about tickets, bills, and clearance are stored in TC's data center. Customers and providers' public information such as virtual IDs, public keys (which are bound to virtual identities), services types, and valid times are also stored in the data center. To thwart ticket duplication and forgery attacks, customers' public key information is updated through a *key update* protocol after each execution of ticket usage. Consequently, our scheme does not require many keys to be stored on the customer side (unlike [22, 31]), where the storage is limited in mobile environments.

In our basic architecture, we do not constraint the communications to be wireless except the one between customer and service provider. Other communications can be carried on any appropriate network systems such as internet. We assume that the devices of customers can be used to generate digital signatures, and those of providers, TCs and TAs are powerful enough to carried out necessary cryptographic operations.

4. Generic Protocol for Controlling Access to Mobile Services

4.1. Preliminaries

Hash function Hash function is also called *one-way function* which is used in cryptographic components such as pseudo-random generation, digital signature and message authentication. Let $H(x)$ be a hash function. For a given y it is computationally hard to find an x such that $H(x) = y$, where x might be a vector. In general, a hash function is a one-way mapping from arbitrary bit-strings $\{0,1\}^*$ to fixed bit-strings $\{0,1\}^k$, and the image values are assumed to be uniformly distribution in $\{0,1\}^k$.

Signature A signature scheme consists of the following algorithms.

- $(SK, PK) = \mathcal{G}en(1^k)$: *Key generation* is a probabilistic polynomial-time algorithm that takes security parameter 1^k as input and outputs a public and secret key pair (SK, PK).
- $\sigma = \mathcal{S}ig(SK, m)$: *Signature signing* is an algorithm that takes (SK, m) as input and outputs a signature σ on message m.
- $\mathcal{V}er(PK, \sigma) \overset{?}{=} 1$: *Signature verification* is a (probabilistic) polynomial-time algorithm that takes (PK, σ) as input and outputs either 1 (True) for acceptance or 0 (False) for rejection.

4.2. The Four Stages of Our Protocol

We explain our generic protocol in four stages – user registration, ticket generation, ticket usage, and key update.

User Registration At the registration stage, U (customer or provider) registers itself with the TA. After authenticates U, TA distributes a secret key SK to U and corresponding public key PK to TC, which are generated by the key generation algorithm: $(SK, PK) = \mathcal{G}en(ID, 1^k)$ (ID is optional in this formula); TA also sends the virtual ID assigned to U along with PK to TC. As mentioned before, the virtual ID can be real ID if the involving party does not care about anonymity. TC stores the virtual ID and public key into its data center. Later, U reads PK from TC data center and verifies the triplet (ID, SK, PK). If (ID, SK, PK) passes the validity check, then the registration process completes successfully. Otherwise, U should report errors and request another registration. This procedure is shown in Fig. 2. For providers, some extra public information, such as the types of services provided and the expiration time by which they are authorized to provide those service, is also published together with their virtual identities and public keys.

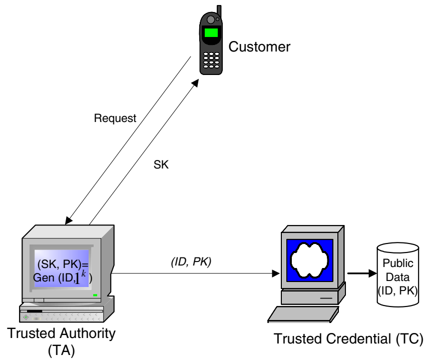


Figure 2. The generic scheme: Registration

Ticket Generation We consider both the case that only one customer is involved in a mobile service transaction and the case that more than one customers are involved in the transaction.

- For the case that only one customer is involved, a ticket can be generated by the customer itself at any time. The major component of ticket is a signature of message m using the customer's secret key SK, i.e., $\sigma = \mathcal{S}ig(SK, m)$, where message m should includes service provider's ID, service type, price and valid time, which are available from the TC's data center and should be consistent with the records.
- For the case that more than one customers are involved, we assume all participants know the set $\{ID_i\}_{i \in Index}$ of virtual identities who are involved. We require that the ticket generation be performed in such way that, each customer signs on m individually (in parallel) and then sends it to a collector. Once receiving all these

sub-signatures, the collector constructs a ticket for all involved customers. The ticket can be used in the same way as in the single user case. The collector could be TC, the provider, one of the customers, or someone else depending on the convenience in practice.

Ticket Usage The ticket usage process is described as following. A customer (or collector) first sends the $Ticket = (m, ID, \sigma)$ to the service provider. The provider first checks whether m has the required format and whether the included information (service provider's ID, service type, price, valid time etc.) is correct. Then the provider reads (m, ID, PK) from the data center of TC and uses it to validate the ticket by computing $\mathcal{V}er(PK, \sigma)$. If false, the provider rejects the service request; otherwise, it provides the service to the customer, signs the ticket to get $Sig_P(Ticket)$ and forwards $Ticket$ along with $Sig_P(Ticket)$ to TC for billing. Note that Sig_P can be any general signature, not necessarily limited by our constraints on customer signatures (see below). $Ticket$ can also be hashed before signing for efficiency in both computation and communication. TC should check whether Sig_P is a valid signature from the provider indicated by m, verify the correctness of $Ticket$ the same way as the provider did, and then make clearance according to the $Ticket$. In multi-users case, (ID, PK) represents the set $\{ID_i, PK_i\}_{i \in Index}$ and $\mathcal{V}er$ is the corresponding verification function (see section 5.2). The ticket usage procedure is illustrated in Fig. 3.

To prevent ticket duplicate (i.e., replay attack), TC updates the customer's public key after each use of a ticket. The corresponding secret key should also be updated on the customer side so that it can be used to generate a new signature, while the previous signature (or ticket) is outdated. The key update issue is addressed in the following subsection.

Key Update One way to update the secret key and public key pair (SK, PK) is to use specific key generation function $\mathcal{G}en$ which satisfies the following condition: there exists a public homomorphic one-way function f, such that $PK = f(SK)$. (In DLP-based schemes, for example, $f : SK \rightarrow g^{SK}$ is such function, where g is a publicly known generator). Then the updated key pair is computed as $(SK', PK') = (SK \cdot m, PK * f(m))$, where "$\cdot$" and "$*$" denote respectively the binary operations in the secret-key and public-key domains. The customer and the provider update secret key and public key respectively using the shared message m. When more users are involved, each key pair is updated independently in the same way.

This update algorithm is vulnerable to *homomorphic forgery attack*. That is, if the signing algorithm $\mathcal{S}ig$ satisfies equation $\mathcal{S}ig(SK_1, m)\mathcal{S}ig(SK_2, m) = \mathcal{S}ig(SK_1 \cdot SK_2, m)$, then after eavesdropping a used ticket $\mathcal{S}ig(SK, m)$, an attacker can easily forge a new ticket $\mathcal{S}ig(SK, m)\mathcal{S}ig(m, m) = \mathcal{S}ig(SK \cdot m, m)$ which is valid under the updated public key $PK' = PK * f(m) = f(SK \cdot m)$. Many existing good signatures such as RSA are insecure against such attack if used in the key update stage. The homomorphic forgery attack enables an attacker to forge a valid ticket, which may lead to type I attack (use-without-pay).

To solve this problem, let the customer concatenate m with corresponding PK (which is obtained from TC) before he signs it. An attacker cannot forge a new ticket $\mathcal{S}ig(SK \cdot m, m||PK')$ from the used one $\mathcal{S}ig(SK, m||PK)$ using homomorphic forgery attack because the contents of the two tickets are different.

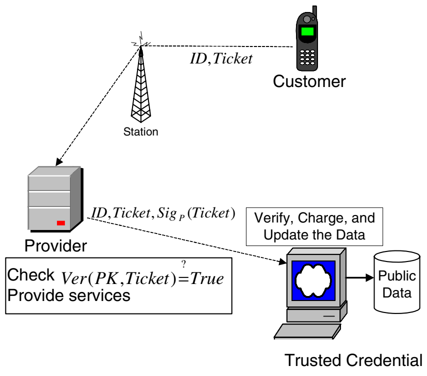


Figure 3. The generic scheme: Ticket usage

4.3. *A Variation of Our Protocol*

An alternative solution to thwart ticket duplicate is to use monotonic $counter$ in the protocol. The $counter$ is stored along with each public key in TC. After each use of a ticket, the TC increments the corresponding $counter$ and publishes it in its data center. In this variation, A customer generates a new ticket by signing a concatenation of message m and corresponding $counter$ (i.e., $\sigma = Sig(SK, m||counter)$) obtained from TC. Neither public key nor secret key needs to be updated. A hacker cannot forge a valid ticket because the content of the ticket is different from the used ones.

Compared with the original form of our protocol, this solution requires slightly more storage in TC for storing the additional $counter$s. The tradeoffs are the convenience and efficiency. In this solution, any concrete signature algorithms can be used. It is also efficient since neither TC nor customers need to update their keys.

5. An Implementation Option

In this section, we present an efficient implementation option of our generic protocol. The implementation option is based on elliptic curve digital signature algorithm (ECDSA). We discuss both tickets involving single users and tickets involving multiple users.

5.1. *Tickets Involving Single Users*

As mentioned in Section 4.2, our generic protocol in its original form can be implemented with any concrete signature schemes satisfying the specified property $PK = f(SK)$, where f is a public homomorphic one-way function. The standard RSA signature [30] is not suitable in this case since it requires $SK \cdot PK = 1 \pmod{\phi}$, where ϕ is secret to TC.

One remedy to this is to choose the signature key pair in the form of $(SK, PK) = (SK, SK^e)$ in registration, where e is the TA's RSA public key, and d the corresponding RSA secret key. After each use of ticket, the customer updates his key as $SK * m$, while the TC updates the corresponding public key as $PK * m^e$.

DLP-based signatures [8,9,12] can be used directly in our protocol. The key pair in these signatures is always of the form $(SK, PK) = (x, g^x)$ for some q ordered element $g \in Z_p^*$, where g is public, and q is a prime factor of $(p-1)$. For instance, Schnorr signature [32], ElGamal signature [12] and digital signature algorithm (DSA) [9] are all suitable. For the same reason, ECDLP-based signatures such as ECDSA [11] can also be employed straightforwardly.

Among these signature algorithms, we recommend to use ECDSA [11] because it is one of the most efficient algorithms. According to [19], the elliptic curve signature is about 12 times faster than Digital Signature Algorithm (DSA) for the same security level.

5.2. Tickets Involving Multi-Users

In multi-users case, a collector constructs a ticket signed by all participants in a group. A straightforward solution is that each participant computes a single signature using its own secret key. The collector constructs the ticket by concatenating a description of the participants and all participant's signatures. More efficient solutions can be obtained by using multi-signatures [3]. Note that most existing multi-signature schemes are complicated; they require that the message should be signed one after another [21,28]. Such sequential process is cumbersome. Multi-signature schemes with the capability of signing the message in parallel would be preferred.

To the best of our knowledge, the only multi-signature scheme with such capability is proposed in [3], which is based on the gap Diffie-Hellman (GDH) group[1]. To thwart the homomorphic forgery attack (in the original form of our protocol), this scheme can be implemented in the following manner.

Let $P = \{P_1, \cdots, P_v\}$ be the set of participants. Let $SK = \{SK_1, \cdots, SK_v\}$ and $PK = \{PK_1, \cdots, PK_v\}$ are their key pairs generated by $\mathcal{G}en(1^k)$ in registration. Let G be a GDH (multiplicative) group of prime order p with $|p| = k$ and let g be a generator of G. Each SK_i is a random element in Z_p^* and PK_i is computed as $PK_i = g^{SK_i}$. Let H be a hash function mapping arbitrary strings to the elements of $G^* = G\ \{1\}$, where 1 denotes the identity element of G. The signing algorithm $\mathcal{MS}ig(m, SK)$ and verification algorithm $\mathcal{MV}er(\sigma, PK)$ are described as follows.

- $\mathcal{MS}ig(SK, m)$: Each participant $P_i \in P$ computes a signature $\sigma_i = H(m||PK)^{SK_i}$ and sends σ_i to the collector. This can be done in parallel. Once the collector receives all v individual signatures, he constructs the multi-signature $\sigma = \prod_{i=1}^{i=v}(\sigma_i)$ and the ticket $Ticket = (m, P, \sigma)$.
- $\mathcal{MV}er(PK, \sigma)$: The verifier computes $PK(P) = \prod_{i=1}^{i=v}(PK_i)$ and then checks whether $(g,\ PK(P), H(m||PK), \sigma)$ is a valid Diffie-Hellman tuple [5]. If it is true, the ticket is accepted, otherwise rejected.

It is obvious that our implementation is secure against the homomorphic forgery attack since each participant signs the concatenation of message m and his current public key. The reason has been analyzed in section 4.2.

[1]The GDH group is known to exist only in some elliptic curves.

6. Discussion and Analysis

6.1. Generic Protocol

The proposed protocol can be implemented based on any practical public-key cryptosystems, such as RSA, ElGamal and ECC (in Section 5, we have discussed an implementation option based on ECDSA). In addition, our protocol can be easily incorporated with different service modes each of which is corresponding to a type of tickets. In [4], Buttyán and Hubaux classified tickets in mobile services into four categories, depending on whether a ticket is bound to customer and provider. In our solution, both customer and provider can choose to be anonymous, thus it allows for all types of tickets. A customer can sell or give his ticket to another customer, or stipulate a range of services to which his ticket applies. The customer may incorporate some usage constraints (e.g., who will provide/accept what kinds of services in which conditions) into a ticket (more precisely, message m) so as to thwart the intercepting or misuse of the ticket. On the other hand, the usage constraints can be published in TC (together with customers' identities and public keys); a service provider can check tickets against such information so as to facilitate fine grain access control in mobile services.

Note that our protocol can be easily integrated with standard encryption techniques and billing methods. Standard encryption can be provided at underlying network layer so that all communications (between TA, TC, customer, and provider) are encrypted. For the billing process, many secure electronic payment protocols such as micro-payment [25, 38] can be applied between TC, customer, and provider. A payment can be prepaid (like telephone card payment) or post-paid (like credit card payment), decided at registration stage.

6.2. Security

We discuss the security aspects of our protocol with respect to type I and type II attacks which were described in section 2.1.

Type I attack (use-without-pay) can be categorized into *duplication*, *forgery*, and *modification* of tickets.

There are two types of duplication attack. The first type is that a customer either uses or transfers a ticket many times (similar to double spending in electronic cash systems). The second type is an eavesdropper, who listens to someone else acquiring a ticket, makes a copy for himself. In our protocol, both are prevented by updating users' public information (public keys or counters) after each use of ticket.

Forgery refers to illegal construction of a valid ticket. Modification means that adversaries modify valid tickets for accessing different services. Our protocol is immune to both since a valid ticket is always associated with a signature signed with a customer's secret key.

To thwart Type II attack (get-paid-without-serve), the signature Sig_P of a service provider is used when the provider forwards a ticket $Ticket$ to TC for charging the bill (payment to TC) and making clearance (payment to the service provider). The signature is used by TC to authenticate the service provider (based on its virtual ID). After this process, both $Ticket$ and $Sig_P(Ticket)$ are stored in TC's data center for a reasonable period of time. In the case of dispute, it is a service provider's responsibility to obtain evidence for having provided services. The evidence can be collected from the customers who have received the services (e.g., signed email confirmation) or from some trusted third parties who are involved in the process of delivering the services (e.g., post office receipt). How to collect such evidence in various applications is outside of the scope of this paper. If a customer figures out that he was charged without receiving the corresponding service, he can report to TC. After verifying the customer's ticket and corresponding service provider's signature, TC requests the service provider to provide the evidence. If the customer's report is proven to be true, TC removes the provider from its service list, reports to the TA for tracing the real identity of the provider, and claims for compensation. Upon receiving the report from TC, TA broadcasts a message to all TCs to suspend or revoke the privileges of the service provider in mobile services.

Not surprisingly, our protocol is subject to denial of service attack in which attackers send a flood of invalid tickets to some service providers. This attack is common in most network systems and can be mitigated at the network layer (e.g., trace and locate attackers with the help of communication service providers).

6.3. Anonymity

Virtual identities, instead of real identities, can be used in all stages of our protocol except the registration stage. A service provider can make its decision on whether to grant service without identifying the real identities of customers. Based on some secure payment protocols, TC can complete the process of charging bills and making clearance without knowing the real identities of users. Therefore, our protocol is immune to both Type III (outsider-to-know-insider) and IV (insider-to-know-insider) attacks provided that TA is well protected. Note that TA is only involved in registration process and in the course of dispute settlement; it may not be difficult to protect TA as it can be put off-line in mobile services.

6.4. Scalability

Traditional solutions for implementing user mobility rely on cross-domain authentication and roaming agreements. A customer, when visiting a foreign domain and accessing a service there, has to authenticate itself to the foreign service provider with the help of its home domain agent. This may involve an authentication process over a long distance, which could be time consuming and expensive. Furthermore, cross-domain authentication requires a foreign service provider to trust the home domain agent. With the rapidly growing of service providers, such schemes will no longer be practical.

Instead of contacting the home domain agent for cross-domain services, a service provider in our protocol verifies tickets. The tickets are generated by customers themselves, with their own secret keys. The ticket acquisition does not require the help of home domain agents or long-distance protocols. Business agreements between different domains may not be needed, though, they might be advantageous in facilitating payments between TCs and service providers.

6.5. Efficiency

The efficiency of a mobile service access protocol can be considered in two aspects. One aspect is the number of rounds for completing a transaction. In our protocol, this parameter is designed to be relatively small. For example, in order to generate a ticket, many ticket based solutions (e.g., [4] and [29]) require a customer to contact with a ticket server or customer care agency, who in turn contacts with a service provider (and possible a certificate authority); the ticket acquisition process may require four to six rounds of communications. In comparison, in our protocol, a customer can generate a ticket by himself; there is no need to contact with other parties in this process. For ticket usage, our protocol requires only one round of communication in which the customer sends his ticket to service provider, while some other solutions require more rounds (e.g., a nonce value is requested to send back and forth between customer and service provider in [4]).

The second aspect is to select an efficient cryptographic algorithm to implement the generic protocol. We have recommended to use ECDSA [11] in section 5.1 because it is one of the most efficient signature algorithms.

7. Related Work

7.1. Comparison with Previous Work

Perhaps the closest work to ours is the mobile service solution proposed by Want et al. [34–36]. The architecture in their solution is similar to ours (they use terms "trusted authentication and registration center" for TA, and "trusted credential" for TC). However, their protocol is different from ours in terms of both security and efficiency. We also note that our protocol is generic, while Wang et al.'s scheme depends on specific signature scheme. In particular, it assumes that TA is equipped with RSA crypto-system in which (n, e) is public and d is secret. To make these differences clear, we summarize Wang et al.'s protocol for single-user tickets first in the following steps. For convenience, we assume that all operations are computed modulo n unless otherwise stated.

Step 1 (Registration to TA): A customer registers to TA by sending his real identity as well as mask identity. After authentication of the customer, TA selects a random number $k \in_R Z_n$ and computes

$$R = k^e, S = k * ID$$

where ID is the mask ID. R and S are sent to the customer through a secure channel. The secret value S and public value $D = S^e$ constitutes the signature key pair of the customer.

Step 2 (Registration to TC): The customer sends (ID, R, D) to TC to setup his public data. The TC validates the triplet by checking

$$D \stackrel{?}{=} R * ID^e$$

If true, TC publishes (ID, D) for signature verification later.

Step 3 (Ticket generation): Let a message m represent a service requirement. A ticket is generated by the customer by signing m: (i) Select random number $r \in_R Z_n$ and compute $T = r^e$. (ii) Compute a hash value $h = H(T||m)$. (iii) Compute a final witness $t = r * (S * m)^{-h}$. (iv) The ticket is (t, T, m). (v) The customer updates his secret S to $S' = S * m$, which will be used for signing a ticket next time.

Step 4 (Ticket usage): When the customer uses the ticket (t, T, m) to require the service, the corresponding service provider first reads (ID, D) from TC, then computes $h = H(T||m)$ and checks whether

$$h \overset{?}{=} H(t^e * D^h * m^{eh}||m)$$

If true, the service provider provides the service; otherwise rejects the request.

Step 5 (Charging and updating): After providing services, the service provider sends the used ticket to TC for charging the bill and updating the public data of the customer. The billing information is stored in TC for the customer to view and access at any time. The public data is updated from (ID, D) to (ID, D') with $D' = D * m^e$. It is clear that $D' = S'^e$; therefore, the ticket generation process and verification processes remain unchanged next time. The old key pair (S, D) cannot be used again.

The protocol uses a specific signing algorithm based RSA keys. Compared with our generic protocol, the scheme is subject to the following attacks.

- *Inconsistent update of keys*: Assume an interceptor captures a ticket sent by a customer to a service provider. The interceptor blocks the ticket so that the service provider cannot receive it. The interceptor then forwards the ticket to TC who will update the public key of the customer according to the scheme. As a result, the customer does not update his secret key. He may resend the ticket, however, it is not valid anymore due to the update of public key. This is a type of denial of service attack. To mitigate this attack, our protocol requires that a digital signature be associated with a service provider's request for update of public key (to thwart replay attack, a monotonic counter can be used in the signature). The provider's signature also helps thwart type II attack (get-paid-without-serve).
- *Bypass of registration*: TA is set up for providing anonymity and resolve confliction. However, in Wang et al.'s scheme, TA is not involved in the initiation of public information. An attacker can easily forge a quadruple (ID', R', S', D') (in the same way as TC does) so that TC can verify $D' = R' * ID'^e$. As a result, TC publishes (ID', D') and the attacker can do whatever a normal customer can do. This attack leads to type I attack (use-without-pay) in which an attacker can impersonate any registered customers. To thwart this attack, our protocol requires that setting up a customer's public data be initiated by TA, instead of the customer. The customer can check the correctness of the public information published by TC.

Now consider Wang et al.'s scheme for multi-user tickets. Assume that a group of v users are going to sign message m. Public data of this group $(ID_1, \ldots, ID_v, g)$ is published in TC, where $g = \Pi_{i=1}^{v} D_i = \Pi_{i=1}^{v} S_i^e$. Each user i in the group generates a share of ticket (t_i, T_i) in the same way as in single-user case, except that $t_i = r_i * (S_i * m)^z$, where z is a prime number publicly known to all signers. A collector collects all (t_i, T_i) and produces a group ticket (t, T, m) by

$$t = \Pi_{i=1}^{v} t_i, T = \Pi_{i=1}^{v} T_i$$

To verify the ticket, TC checks

$$T \overset{?}{=} t^e * g^{-z} * m^{-zve}$$

This scheme is subject to type I attack (use-without-pay). An attacker can impersonate any user, say user 1, using public information z, e. In particular, he first selects a random number $t_1 \in_R Z_n^*$, and then computes $T_1 = t_1^e * D_1^{-z} * m^{-ze}$. If all other ticket shares (t_i, T_i) are correct, the group ticket (t, T, m) can be verified because

$$\begin{aligned} T = \Pi_{i=1}^{v} T_i = T_1 * \Pi_{i=2}^{v} T_i \quad &= \quad (t_1^e * D_1^{-z} * m^{-ze}) * \Pi_{i=2}^{v} t_i^e * (S_i * m)^{-ze} \\ = (t_1^e * D_1^{-z} * m^{-ze}) * \Pi_{i=2}^{v} t_i^e * D_i^{-z} * m^{-ze} \quad &= \quad \Pi_{i=1}^{v} t_i * \Pi_{i=1}^{v} D_i^{-z} * m^{-zve} \\ = t^e * g^{-z} * m^{-zve} & \end{aligned}$$

In addition, the attacker can forge a group ticket by himself using public information z, v, e, g. In particular, he chooses a random value $t \in_R Z_n^*$ and computes $T = t^e * g^{-z} * m^{-zve}$. It is clear that (t, T, m) is a valid ticket.

7.2. Account-Based Versus Ticket-Based

In general, mobile payment methods can be classified into account-based and ticket-based (or token-based). In traditional account-based approach (e.g., [10, 22]), each customer is associated with an account that is maintained by an service provider. It may require cross-domain authentication for mobile services due to the need of accounting and charging mechanisms for costs to be recovered by the foreign domain.

Compared to account-based solution, ticket-based approach is more flexible in that customers can easily construct personalized service profiles by buying an appropriate set of tickets or generating tickets themselves. An early work in this area is described in [29]. In that work, the concept of ticket is similar to that of tradition ticket (e.g., flight ticket) that people are familiar with. The ticket is bought by a customer from a ticket server through mutual authentication, in which process the ticket server may need to contact with a service server (like service provider in our scheme) and a certification authority. The whole ticket acquisition process may require four to six rounds of communications, which may not be very convenient in mobile services.

7.3. Anonymity in Mobile Services

The problem of anonymous yet accountable service access in mobile communication systems is addressed in [2]. The authors proposed the use of an anonymous tokening system, similar to anonymous digital cash systems, to solve this problem. Tokens are issued by the home network of the user. In order to provide anonymity, the tokens are blinded and paid at acquisition time. The disadvantage of this solution is that it may not always be convenient for a user to obtain tokens from his home network in mobile context. In [16, 25], the tokens are generated by customers. These solutions have the weakness of not providing anonymity for customers with respect to service providers. Later, Buttyán and Hubaux introduced a framework that allows anonymous access to services in mobile communication systems [4]. However, no protocol was proposed for the interactions among customers, service providers, and customer care agency (somehow like TC and TA in our scheme). In [34–36], Wang et. al. proposed a few ticket-based protocols for anonymous access to mobile services; however, these protocols are vulnerable to ticket forgery attacks as discussed in above section 7.1.

7.4. Other Related Work

In tradition, cross domain authentication has been used to facilitate user mobility between domains. An example is Kerberos [33] which relys on inter-realms interaction before issuing a ticket in a different realm. Another example is Shibboleth [6] which is a recent inter-realm access control solution for research and education. In these solutions, a user has a home domain that authenticates the user to foreign domains. Not only this may require long distance authentication process, which could be potentially time consuming and expensive, but also requires trust relationship to be established between foreign domain and home domain. This kind of architecture works well for relative few service providers and services. It is arguable that it is a ideal solution in an environment where there are many diverse services.

8. Conclusion

With the widespread use of advanced mobile devices and the availability of wireless networking infrastructure, there has been considerable growth in mobile services [15]. In this paper, we first classified the attacks on mobile services and identified the desirable properties of mobile service architecture as well as related protocols. Following that, we proposed a ticket-based secure architecture and a generic protocol for controlling access to mobile services. An efficient implementation option of the generic protocol, which is based on elliptic curve digital signature algorithm, was also discussed. Detailed analysis shows that our work not only enhances the security and anonymity, but also provides high flexibility (i.e. compatible with various cryptographic algorithms and service modes), scalability, and efficiency in mobile services. Possible future work includes full-fledged implementation of the proposed solution and extension to other applications such as secure grid computing.

References

[1] Wael Adi, Ali Al-Qayedi, Abdulkarim Al Zarooni, and Ali Mabrouk. “Secured Multi-Identity Mobile Infrastructure and Offline Mobile-assisted Micro-payment Application”. In proceedings of WCNC 2004, 2004.

[2] B. Askwith, M.Merabti, Q. Shi, and K.Whiteley. “Achieving user privacy in mobile networks”. In proceedings of the 13th Annual Computer Security Applications Conference, 1997.

[3] A. Boldyreva, “Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme," in *Public Key Cryptography - PKC 2003*, LNCS 2567, pp. 31ĺC46. Springer-Verlag, 2003.

[4] L. Buttyan, and J. Hubaux, “Accountable anonymous access to services in mobile communication systems," in *Proc. 18th Symp. Reliable Distributed Systems*, Lausanne, Switzerland, Oct. 1999, pp. 384-389.

[5] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, “ Aggregate and verifiably encrypted signatures from bilinear maps," in *Proceedings of Eurocrypt ’03*, LNCS 2656, pp. 416-432, 2003.

[6] S. Carmody, “Shibboleth Overview and Requirements," http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-requirements-01.html, February, 2001

[7] D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash," in *Advances in Cryptography - Crypto'88*, LNCS 403, Springer-Verlag, 1990, pp. 319-327.

[8] W. Diffie and M. E. Hellman, "New Direction in Cryptography," *IEEE Transactions on Information Theory*, 22(6), pp. 644-654, 1976.

[9] FIPS 186-1, "Digital Signature Standard (DSS)," National Institute for Standards and Technology, 1998.

[10] Yuanjun Dai and Lihe Zhang. "A Security Payment Scheme of Mobile e-Commerce". In proceedings of ICCT'03, 2003.

[11] ANSI X9.62/9.63, "The Elliptic Curve Digital Signature Algorithm (ECDSA)," American National Standards Institute, 1998.

[12] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms", *IEEE Trans. Info. Theory* IT-31(4), pp. 469-472, 1985.

[13] D. Ferraiolo, J. Barkley, and D. Kuhn, "Role-based access control model and reference implementation within a corporate intranet," *ACM Trans. Information and System Security*, vol. 2, no. 1, pp. 34-64, 1999.

[14] Alia Fourati, Hella Kaffel Ben Ayed, Farouk Kamoun, and Abdelmalek Benzekri. "A SET Based Approach to Secure the Payment in Mobile Commerce". In proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN'02), 2002.

[15] Ali Grami, and Bernadette H. Schell, "Future Trends in Mobile Commerce: Service Offerings, Technological Advances and Security Challenges," in Proceedings of the 2nd Annual Conference on Privacy, Security and Trust, 2004.

[16] G. Horn, and B. Preneel, "Authentication and payment in future mobile systems," in *Proc. Euro. Symp. Research in Computer Security*, LNCS 1485, Springer-Verlag, 1998, pp. 277-293.

[17] G. Horn, and B. Preneel, "Authentication and payment in future mobile systems," *J. Computer Security*, vol. 8, pp. 183-207, 2000.

[18] Zheng Huang and Kefei Chen. "Electronic Payment in Mobile Environment". In proceedings of the 13th International Workshop on Database and Expert Systems Applications (DEXA'02), 2002.

[19] N. Koblitz, "An elliptic curve implementation of the finite field digital signature algorithm," in *Proc. of Crypto '98*, LNCS 1462, pp. 327-737, Springer-Verlag Berlin Heidelberg 1998.

[20] Mi-Ae Kim, Han-Ki Lee, Seong-Whan Kim, Won-Hyoung Lee, and Eung-Kwan Kang. "Implementation of Anonymity-Based e-Payment System for M-Commerce". 2002.

[21] K. Kawauchi and M. Tada, "On the Exact Security of Multi-signature Schemes Based on RSA," in *Information Security and Privacy - ACISP'03*, LNCS 2727, pp. 336-349. Springer-Verlag, 2003.

[22] Supakorn Kungpisdan, Bala Srinivasan, and Phu Dung Le. "A Secure Account-based Mobile Payment Protocol". In proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'04), 2004.

[23] A. Lubinski, "Security issues in mobile database access," in *Proc. IFIP WG 11.3 12th Int. Conf. Database Security*, Chalkidiki, Greece, 1999, pp. 223-234.

[24] A. Mehrotra, and L. Golding, "Mobility and security management in the GSM system and some proposed future improvements," *Proc. IEEE*, vol.86, pp.1480-1496, July 1998.

[25] K. Martin, B. Preneel, C. Mitchell, H. Hitz, A. Poliakova, and P. Howard, "Secure billing for mobile information services in UMTS," in *Proc. 5th Int. Conf. Intelligence Services and Networks - IS&N'98*, LNCS 1430, Springer-Verlag, 1998, pp.535-548.

[26] Gianluigi Me. "Security overview for m-payed virtual ticketing". In proceedings of the 14^{th} IEEE International Symposium on Personal, Indoor, and Mobile Radio Communication (PIMRC'03), 2003.

[27] Masashi Morioka, Yoshifumi Yonemoto, Takashi Suzuki, and Minoru Etoh. "Scalable Security Description Framework for Mobile Web Services". In proceedings of ICC'03, 2003.

[28] K. Ohta and T. Okamoto. "Multi-signature schemes secure against active insider attacks." In

IEICE Trans. Fundamental, vol. E82-A, No. 1, pp. 21-31, January 1999.

[29] B. Patel, and J. Crowcroft, “Ticket based service access for the mobile user," in *Proc. Int. Conf. Mobile Computing Networking - MobiCom’97*, Budapest, Hungary, Sept. 1997, pp. 223-232.

[30] R. L. Rivest, A. Shamir, and L. M. Adleman, “A method for obtainig digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120-126, Feb 1978.

[31] R. Rivest and A. Sharmir. “PayWord and MicroMint: two simple micropayment schemes”. In proceedings of 1996 International Workshop on Security Protocols, LNCS 1189, pp. 69-87, 1996.

[32] C.P. Schnorr, “Efficient identification and signatures for smart cards," in *Proc. of Crypto’89,* LNCS 435, pp. 239-252. Springer-Verlag, 1990.

[33] J. Steiner, C. Neuman, and J. Schiller, “Kerberos: An Authentication Service for Open Network Systems," Proceedings of the 1988 Winter USENIX Conference, pages 191-202, Feburary 1988.

[34] H. Wang, J. Cao, Y. Zhang, “Ticket-based service access scheme for mobile users," in *Proc. 25th Australian Computer Science Conference - ACSC 2002*, Australian Computer Society, Vol. 4 pp. 285-292, Monash University, Melbourne, Victoria, 2002.

[35] H. Wang, Y. Zhang, J. Cao, and V. Varadharajan, “Achieving secure and flexible M-services through tickets," *IEEE Trans. Systems, Man, and Cybernetcis - Part A: Systems and Humans*, vol. 33, no.6, pp. 697-708, November 2003.

[36] H. Wang, L. Sun, Y. Zhang, and J. Cao, “Anonymous access scheme for electronic-services," in *Proc. 27th Australasian Computer Science Conference - ACSC 2004*, Australian Computer Society, Vol. 26, pp. 296-305. Dunedin, New Zealand, 2004.

[37] U. Wilhelm, S. Staamann, and L. Buttyan, “On the problem fo trust in mobile agent systems," in *Proc. IEEE Network Distributed Systems Security Symp. - NDSS’98*, San Diego, CA, 1998, pp. 11-13.

[38] H. Wang and Y. Zhang. “Untraceable off-line electronic cash flow in e-commerce. In proceedings of 24th australian computer science conference, pp. 191-198, IEEE Computer Society, 2001.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Use of XACML Policies for a Network Access Control Service

Gabriel López [a], Óscar Cánovas [b] and Antonio F. Gómez-Skarmeta [a]
[a] *Department of Information and Communications Engineering*
[b] *Department of Computer Engineering*
University of Murcia, Spain

Abstract. The interest in policy specification languages is increasing thanks to the proliferation of authorization solutions that need to define their resource access policies by means of them. These solutions define their own policy syntax, usually based on XML, which involves the definition of non-interoperable policies and non-heterogeneous environments. XACML has been defined with that purpose and is getting more and more acceptance for those type of environments as a valid alternative to proprietary policies. In this paper, we present the definition of the whole policies set needed in an authorization scenario, specifically, the NAS-SAML, which defines a network access control service based on SAML and the AAA architecture. We present the XACML documents representing those policies and the entities involved in the their management life cycle.

Keywords. XACML, Policy, Authorization, AAA, SAML

1. Introduction

During the last years, we have experienced an increasing interest in policy specification languages, not only for security purposes but also for network management, workflow models, etc. Focusing on security scenarios, the proliferation of authorization architectures, such as the NAS-SAML proposal [1], the PERMIS project [2], the Akenti system [3], or Shibboleth [4], evidences the relevance of using appropriate policy languages to define accurate access control policies. In fact, every authorization solution needs a way to express how the authorization data is managed, that is, how users use their attributes to gain access to protected resources. Generally, the different authorization solutions define their own policy syntax, usually based on XML, which implies the proliferation of non-interoperable policies and the absence of heterogeneous environments.

Therefore, it seems that a standard schema to represent that set of policies is required. XACML (eXtensible Access Control Markup Language) [5], the OASIS proposal for a standard access control language, was defined to solve this problem. XACML is XML-based and includes two different specifications: the first one is an access control policy language, which defines the set of subjects that can perform particular actions on a subset of resources; the second one is a representation format to encode access control requests and responses, that is, a way to express queries about whether a particular access should be allowed and the related answers. This paper describes the use of XACML as the policy language that we used to define the set of policies of a network access control scenario [1]. The scenario defines a network access control service based on the AAA architecture and SAML, hereafter the NAS-SAML solution.

The rest of this paper is structured as follows. Section 2 defines the NAS-SAML scenario. Then, Section 3 defines the set of policies required and Section 4 gives examples about how those policies can be expressed by means of XACML. Section 5 describes the relationship among the different components of the scenario and the policies. Section 6 describes the related work that informed our research. Finally, we conclude the paper with our remarks and future directions.

2. SAML-based Network Access Service

In [1], we present a network access control approach based on X.509 identity certificates and authorization attributes, which addresses some of the challenges derived from the integration of existing authentication systems and a flexible, scalable and manageable authorization system. The proposal is based on the SAML and the XACML standards, which are used for expressing access control policies based on attributes, authorization statements, and authorization protocols. The starting point is a network scenario based on the 802.1X standard [6] and the AAA (Authentication, Authorization and Accounting) architecture [7], where we centralize all the operations related to authentication, authorization, and accounting.

In [8], we also propose a solution to the integration of the NAS-SAML scenario with other authorization systems, such as PERMIS. This integration is based on the definition of the different cooperating elements, and the specification of the policies able to translate source authorization credentials, e.g. X.509 attribute certificates [9], into SAML statements.

2.1. System Architecture

The system operates as follows. Every end user belongs to a home domain, where he was given a set of attributes stating the roles he plays. When the end user requests a network connection in a particular domain by means of a 802.1X connection, the request is obtained by the AAA server, and it makes a query to obtain the attributes linked to the user from an authority responsible for managing them (governed by a *Role Assignment Policy*). When the user's home domain is based on a non-SAML authorization system, the AAA server uses a credential conversion service, as defined in [8], to translate the user's authorization credentials into SAML statements, using the *Conversion Policy*. Finally, the AAA server sends an authorization query to a policy decision point and, depending on the *Resource Access Policy*, the response encodes whether the attributes satisfy the policy. Furthermore, it also establishes the set of obligations derived from the given decision, for example QoS properties, security options, etc. This general scheme works both in single and inter-domain scenarios, and using push an pull based communications.

2.2. Policy Requirements

This section describes the main policies involved in this scenario: *Role Assignment Policy*, *Resource Access Policy* and *Conversion Policy*, its requirements and the relationships among them.

The *Role Assignment Policy* is located in every home domain where the assignment between user and attributes is defined. The entity controlling this assignment is called the *Source Authority or SA*. Figure $1_{a)}$ shows a high level view of this policy. When an end user wants to obtain his attributes, or those attributes are requested by another entity controlling a protected resource, a request message including the user's subject, and other additional information, has to be sent to the SA. When the SA receives the request, it may enable the assignment of one or more of those attributes to the user in a static way, for example, depending on the information contained in a LDAP repository, or it can do it through a more elaborated policy, taking into account the resource and the attributes that can be revealed to the different foreign domains. For example, a university member has several attributes assigned, such as *professor* or *researcher*, and he would like to use the first one when accessing the *public network* and the second one when accessing the *laboratory network*.

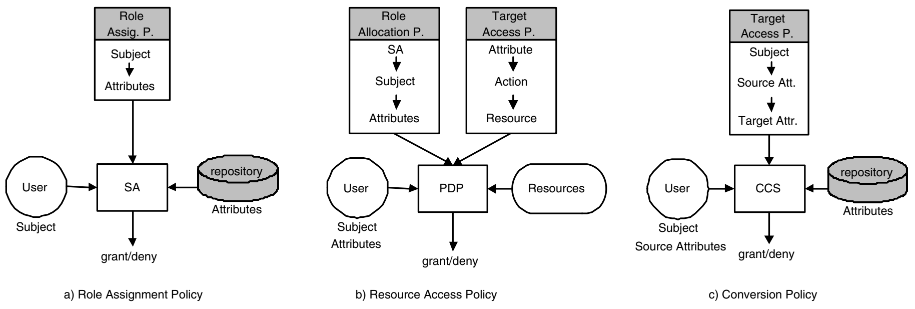


Figure 1. Policies relationship

Resource Access Policy is the policy defined to protect the target resources. When a user wants to get access to a resource, he has to present his credentials to the entity controlling that resource, the *Policy Decision Point or PDP*. The user, in some way, specifies his name, the set of attributes assigned by the *Role Assignment Policy*, and the source domain where he comes from. This policy decides not only whether the attributes involve the permission to execute an action over the desired resource, but also, if the user from that home domain has permissions to hold those attributes, in a similar way as defined in other solutions [2]. These two conditions are specified by two different policies, the *Target Access Policy*, which defines the access rules, and the *Role Allocation Policy*, which recognizes the authority of the existing SAs. For example, the *Target Access Policy* may specify that a user holding the attribute *student* can *access* to a service between *8:00AM and 17:00PM*. The *Role Allocation Policy* would recognize that users from domain *University A* can present the attribute *student*. Figure $1_{b)}$ shows how this policy interacts with the rest of components.

The *Conversion Policy*, used in multi-domain scenarios where the user's home domain is based on a different authorization scheme to the one proposed in NAS-SAML, translates authorization credentials, e.g. Attribute Certificates, into SAML Attribute statements. This policy, managed by the *Credential Conversion Service or CCS*, contains information about the valid home domains, recognized authorization attributes and conversion rules to generate the SAML sentences. For example, the students from *University A* have obtained X.509 ACs including the attribute *student*. When those students try to get access to protected resources in *University B*, which is based on the

NAS-SAML system, the *Conversion Policy* might specify that ACs holding the attribute *type=university:role, value=student*, have to be translated into the SAML attribute *type=university:role, value=foreign-student*. Figure $1_{c)}$ shows a high level view of this policy.

3. Policies Definition

This section defines the different policies and how they can be used in a network access control scenario.

3.1. Role Assignment Policy

This policy, as we commented before, is managed by a SA who, after receiving credential queries from end users or external domains, decides which user's attributes or roles must be assigned and/or disclosed. Every element of this policy is composed by the following set of objects:

- *Subjects*: Users allowed to hold the different attributes. In the proposed scenario, they are defined using X.500 subtrees.
- *Attributes*: Roles assigned to the subjects. They define the role(s) they play in the organization. Attributes are defined by a type and value pair.
- *Conditions*: This element is optional, and can be used to define conditions in the assignment and/or disclosure process. In the NAS-SAML scenario, these conditions might define that users from a home domain will be enabled to hold their roles only at specific time slots.

An example of *Role Assignment Policy* in a university environment is when users pertaining to the X.500 subtree *Subject="ou=Students,ou=Computer Science,o=University A,c=C"* are allowed to hold the role *Attribute="Student"*, but when *Conditions="from 8:00AM to 20:00PM"*.

3.2. Role Allocation Policy

This policy, located in the target domain where the user requests the network access, recognizes the authority of the different SAs to assign attributes to end users. Every recognition rule has the following elements:

- *Attributes*: type and value of the attributes or roles the user presents. If null, it means users can present any attribute.
- *Subjects*: Users allowed to hold those attributes. If null, it means any user from the specified Source Authorities can present attributes. In the proposed scenario, subjects will be defined as X.500 DNs.
- *Source Authorities*: authorities assigning the attributes. If null, it means the authority is not taken into account in the decision process. It is represented by a X.500 DN.
- *Constraints*: Set of constraints imposed to that recognition.

For example, the policy could express that the set of subjects defined by the X.500 subtree *Subjects="ou=Student,ou=Computer Science,o=University A,c=C"*, managed by *Source Authority="o=University A,c=C"*, have permissions to hold the role *Attribute="Student"*, but only *Conditions="from the 8:00AM to the 17:00PM from Monday to Friday"*.

3.3. Target Access Policy

The *Target Access Policy* comprises a set of *target access* elements. Each of them grants an initiator with a specified set of roles the permission to carry out the specified actions on the specified list of targets, but only if the conditions specified are true. Every target access element has the following elements:

- *Attributes*: Set of allowed attributes or roles to execute the actions on the resource. If null, it means any role has permission to perform the specified action on this resource.
- *Resources*: Set of controlled resources. It must be at least one resource element. In NAS-SAML, the resource is the network identifier.
- *Actions*: In NAS-SAML the default action is *enable*.
- *Conditions*: Users holding some of the attributes have permission to execute some actions on the specified resources only if the conditions are fulfilled. Otherwise, the permission will be denied. Those conditions can establish time constraints or other constraints related to contextual information, as the network workload.
- *Obligations*: Once the action has been granted, some obligations defining network properties might be applied depending on the user's role(s). Obligations can specify options related to network addressing, security or QoS properties that must be enforced.

For example, this policy might express that *Role="Student"* can gain access to *Resource="wireless-network"* under *Conditions="from 8:00AM to 18:00PM*, and with *Obligations="IPv6 address from range 2001:720:1710::/64"*.

3.4. Conversion Policy

The *Conversion Policy* is used to translate the user's home attributes into the format required in the target domain. It defines the following elements:

- *Subject*: One or more identifiers specifying valid home domains. It is represented as X.500 DN subtrees.
- *Source Attributes*: Set of credentials expressed as type and value pairs.
- *Target Attributes*: Corresponding user attributes (type and value) used in the target domain according to the internal authorization scheme, for example, SAML attributes.

For example, supposing a home domain based on X.509 ACs and a SOA *Subject="cn=SOA,o=University B,c=C"*, the user's AC can contain the attribute *Source Attr.="oid=2.3.4.5; value=Student"*. This attribute could be translated into *Target Attr.="type=target:domain:role; value=Foreign-Student"*.

4. XACML Policies definition

This section defines a way to use XACML to meet the requirements imposed by the NAS-SAML scenario. It includes the definition of a *Role Assignment Policy*, a *Conversion Policy*, and the policy for a hierarchical role based access control (RBAC) model, the *Resource Access Policy*.

The main element of all XACML policies is a *Policy* or *PolicySet* element. A *PolicySet* is a container that can hold other *Policies* or *PolicySets*, as well as references to other policies (*PolicyIDReference*). A *Policy* represents a single access control policy, expressed by a set of *Rules*. A *Policy* or *PolicySet* may contain multiple policies or *Rules*, each of which may evaluate to different access control decisions. XACML needs some way of reconciling the decisions each makes, and this is done through a collection of *Combining Algorithms*. An example of those is the *Permit Overrides Algorithm*, which says that if at least an evaluation returns *Permit*, then the final result is also *Permit*. A Policy or PolicySet element may also specify a set of *Obligation Attributes* that will be returned to the PEP to be applied, if the element is evaluated as *Permit*.

XACML provides another feature called a *Target*. It is a set of simplified conditions for the *Subject*, *Resource* and *Action* that must be met for a *PolicySet*, *Policy* or *Rule* to apply to a given request. If all the conditions of a *Target* are met, then its associated PolicySet, Policy, or Rule applies to the request. Once a *Policy* is found its rules are evaluated. The main element of a rule is the *Condition*. If the *Condition* evaluates to true, then the *Rule's Effect* (Permit or Deny) is returned.

The main object that XACML deals in is attributes. Attributes are named values of known types. Specifically, attributes are characteristics of the Subject, Resource, Action, or Environment in which the access request is made. In the proposed policies, roles are expressed as XACML *Subject Attributes* or *Resource Attributes*, depending on the policy.

4.1. Role Assignment Policy

The *Role Assignment Policy* can be expressed by means of a *Policy* element that contains the assignment rules. Every *Rule* is composed by a *Target* element including the set of subjects, the set of attributes or roles assigned to the subjects, the default action, *enable*, and the conditions of this assignment.

Figure 2 shows an example of this policy. As we can see, the subjects specified by *ou=Computer Science,o=University A,c=C* are able to use the role type *role-id*, with value *Student*, but only *from 9AM to 7PM*.

4.2. Resource Access Policy

This policy is based on the non-normative RBAC profile defined in [10]. Due to the requirements imposed by the NAS-SAML scenario and the *Resource Access Policy*, this profile has been adapted in two ways. The first one is the definition of the *Role Allocation Policy* which differs from the structure defined in [10] since we needed to include some information about the user's home domain. The second one is the behavior of the *Context Handler* element, as is described in Section 5.

```
<Policy PolicyId="SA_RoleAssignment_Policy">
 <Target/>
 <Rule Effect="Permit" RuleId="student-rule-id">
  <Target>
   <Subjects>
    <Subject>
     <SubjectMatch>
      <AttributeValue>ou=Student,
            ou=Computer Science,o=University A,c=C
      </AttributeValue>
      <SubjectAttributeDesignator AttributeId="subject-id"/>
     </SubjectMatch>
    </Subject>
   </Subjects>
   <Resources>
    <Resource>
     <ResourceMatch>
      <AttributeValue>Student</AttributeValue>
      <ResourceAttributeDesignator AttributeId="role-id"/>
     </ResourceMatch>
    </Resource>
   </Resources>
   <Actions>
    <Action>
     <ActionMatch>
      <AttributeValue>enable</AttributeValue>
      <ActionAttributeDesignator AttributeId="action-id"/>
     </ActionMatch>
    </Action>
   </Actions>
  </Target>
  <Condition>
   <Apply FunctionId="function:and">
    <Apply FunctionId=":function:time-greater-than-or-equal">
     <Apply FunctionId="function:time-one-and-only">
          <EnvironmentAttributeDesignator AttributeId=":current-time""/>
          <AttributeValue>9h</AttributeValue>
     </Apply>
    </Apply>
    <Apply FunctionId=":function:integer-less-than-or-equal">
     <Apply FunctionId=":time-one-and-only">
          <EnvironmentAttributeDesignator AttributeId=":current-time"/>
          <AttributeValue>19h</AttributeValue>
     </Apply>
    </Apply>
   </Apply>
  </Condition>
 </Rule>
</Policy>
```

Figure 2. Role Assignment Policy example

4.2.1. *Role Allocation Policy.*

This policy can be expressed by means of a *PolicySet* element, which includes a *Policy* element for every SA that is recognized in the target domain. This policy defines the SA's subjects, by means of a *Subject Attribute*, in a high level *Target* element. It also defines the set of *Rules* for every subject recognized by that SA. Each *Rule* is composed by the set of *Subject Attributes* specifying the user's subjects, the attributes or roles permitted to those users as a set of *Resource Attributes*, and the default *Action*, *hasPrivilegesOfRole*. In this policy, the SA and user's subjects are defined as *Subject Attribute* elements, using different attribute identifiers.

```
<PolicySet PolicySetId="PDP_Role_Allocation_Policy">
 <Target/>
  <Policy PolicyId="RoleAssignment_1">
   <Target>
    <Subjects>
     <Subject>
      <SubjectMatch>
           <AttributeValue>cn=SA,o=University A,c=C</AttributeValue>
           <SubjectAttributeDesignator AttributeId="sa-id"/>
      </SubjectMatch>
     </Subject>
    </Subjects>
   </Target>
   <Rule Effect="Permit" RuleId="rule-id">
   <Target>
    <Subjects>
     <Subject>
      <SubjectMatch>
           <AttributeValue">ou=Student,ou=Computer Science,
                          o=University A,c=C</AttributeValue>
           <SubjectAttributeDesignator AttributeId=":subject-id"/>
      </SubjectMatch>
     </Subject>
    </Subjects>
    <Resources>
     <Resource>
      <ResourceMatch>
           <AttributeValue>Student</AttributeValue>
           <ResourceAttributeDesignator AttributeId=":resource-id"/>
      </ResourceMatch>
     </Resource>
    </Resources>
    <Actions>
     <Action>
      <ActionMatch>
           <AttributeValue>hasPrivilegesOfRole</AttributeValue>
           <ActionAttributeDesignator AttributeId=":action-id"/>
      </ActionMatch>
     </Action>
    </Actions>
   </Target>
   <Condition>
    <Apply FunctionId=":function:and">
     <Apply FunctionId=":function:time-greater-than-or-equal">
      <Apply FunctionId=":function:time-one-and-only">
           <EnvironmentAttributeDesignator AttributeId=":current-time"/>
           <AttributeValue>9h</AttributeValue>
      </Apply>
     </Apply>
     <Apply FunctionId=":function:integer-less-than-or-equal">
      <Apply FunctionId=":function:time-one-and-only">
           <EnvironmentAttributeDesignator AttributeId=":current-time"/>
           <AttributeValue>19h</AttributeValue>
      </Apply>
     </Apply>
    </Apply>
   </Condition>
  </Policy>
</PolicySet>
```

Figure 3. Role Allocation Policy example

Figure 3 shows an example of how this policy can be expressed. This example defines the policy element stating the roles that can be allocated by the SA *cn=SA,o=University A,c=C*. For this SA, the policy contains a rule specifying that subjects *ou=Student, ou=Computer Science,o=University A,c=C* can be assigned to the *Student* role.

4.2.2. Target Access Policy.

In order to represent this policy we need to define the user's role as the *Subject Attribute* component in a global *Target* element of the main *PolicySet*, called *TargetAccessPolicy*. Then, this policy references the permission policy associated to this role, as described in [10]. This permission policy specifies the set of permission, defined as *Rules*, associated to this role. Hierarchical role permissions can be defined using the *PolicySetIDReference* in the permission policy, pointing to the policy containing the inherited permissions.

```
<PolicySet PolicySetId="TargetAccessPolicy" >
 <Target/>
 <PolicySet PolicySetId="TargetStudentPolicy" >
  <Target>
   <Subjects>
    <Subject>
     <SubjectMatch>
          <AttributeValue>Student</AttributeValue>
          <SubjectAttributeDesignator AttributeId="role-id"/>
     </SubjectMatch>
    </Subject>
   </Subjects>
  </Target>
  <PolicySetIdReference>Permission:Role:Student
  </PolicySetIdReference>
 </PolicySet>
</PolicySet>

<PolicySet PolicySetId="Permission:Role:Student">
 <Target/>
 <PolicySet PolicySetId="PPS:Student">
 <Target/>
  <Policy PolicyId="Permission_1">
  <Target/>
  <Rule RuleId="ruleid" Effect="Permit">
   <Target>
    <Resources>
     <Resource>
      <ResourceMatch>
          <AttributeValue>wireless-network
                         </AttributeValue>
          <ResourceAttributeDesignator
                         AttributeId=":resource-id"/>
      </ResourceMatch>
     </Resource>
    </Resources>
    <Actions>
     <Action>
      <ActionMatch>
          <AttributeValue>enable</AttributeValue>
          <ActionAttributeDesignator AttributeId="action-id"/>
      </ActionMatch>
     </Action>
    </Actions>
   </Target>
  </Rule>
  <Obligations>
   <Obligation ObligationId="obligation-id" FulfillOn="Permit">
    <AttributeAssignment AttributeId="Framed-IPv6-Prefix">
          2001:720:1710:100::/64</AttributeAssignment>
   </Obligation>
   <Obligation ObligationId="obligation-id" FulfillOn="Permit">
    <AttributeAssignment AttributeId="Framed-Type">
          VLAN</AttributeAssignment>
   </Obligation>
   <Obligation ObligationId="obligation-id" FulfillOn="Permit">
    <AttributeAssignment AttributeId="Framed-Medium-Type">
          IPv6</AttributeAssignment>
   </Obligation>
   <Obligation ObligationId="obligation-id" FulfillOn="Permit">
    <AttributeAssignment AttributeId="Tunnel-Private-Group-Id">
          100</AttributeAssignment>
   </Obligation>
   <Obligation ObligationId="obligation-id" FulfillOn="Permit">
    <AttributeAssignment AttributeId="QoS-Filter-Rule">
          Class1</AttributeAssignment>
   </Obligation>
  </Obligations>
 </Policy>
 </PolicySet>
 <PolicySetIdReference>Permission:Role:Member
 </PolicySetIdReference>
</PolicySet>
```

Figure 4. Target Access Policy example

Figure 4 shows an example of the *Target Access Policy*. In this figure we can see two root *PolicySet* elements. The first one defines the role type and value, in this case *role-id* and *Student*. It contains a reference to a permission policy, which defines that the *wireless-network* resource can be *enabled* by this role.

This example also shows the set of obligations derived from this decision. That is, if a user holding the *Student* role, requests *wireless-network* connection, it should obtain an IPv6 address from the range *2001:720:1710:100::/64* and should be located in the *VLAN 100*. Moreover, the network must guarantee a quality of service established by *Class1* type. Finally, we can see how the role *Student* inherits the permissions from the role *Member*.

4.3. Conversion Policy

Figure 5 shows a simple *Conversion Policy* composed by the set of policies related to every home domain. The main *PolicySet* element specifies the *translate* action, which will be common for all the policies. Then, a *PolicySet* element is defined by every domain that needs to translate attributes into SAML sentences. The policy for every domain defines another policy for every attribute that needs to be translated. This is due to the use we make of the *Obligation* element.

```
<PolicySet PolicySetId="GlobalConversionPolicy">
  <Target>
    <Actions>
     <Action>
      <ActionMatch>
        <AttributeValue>translate</AttributeValue>
         <ActionAttributeDesignator  AttributeId="..:action"/>
      </ActionMatch>
     </Action>
    </Actions>
  </Target>
  <PolicySet PolicySetId="ConversionPolicy">
    <Target>
           <Subjects>
              <Subject>
               <SubjectMatch>
                 <AttributeValue>o=AC-based domain,c=C</AttributeValue>
                 <SubjectAttributeDesignator AttributeId="...:external:SOA"/>
               </SubjectMatch>
              </Subject>
           </Subjects>
    </Target>
    <Policy PolicyId="SimplePolicy1">
     <Target/>
      <Rule RuleId="SimpleRule1" Effect="Permit">
       <Target>
        <Resources>
            <Resource>
              <ResourceMatch>
                <AttributeValue>studentRole</AttributeValue>
                <ResourceAttributeDesignator      AttributeId="...:resource-id"/>
              </ResourceMatch>
              <ResourceMatch>
                <AttributeValue>ERASMUS</AttributeValue>
                <ResourceAttributeDesignator      AttributeId="...:value"/>
               </ResourceMatch>
            </Resource>
        </Resources>
      </Target>
     </Rule>
     <Obligations>
      <Obligation ObligationId="Obligation1" FulfillOn="Permit">
       <AttributeAssignment AttributeId="urn:role:student">ERASMUS
            </AttributeAssignment>
      </Obligation>
     </Obligations>
     </Policy>
   </PolicySet>
</PolicySet>
```

Figure 5. Conversion Policy example

For example, *GlobalConversionPolicy* defines the whole set of attributes that can be translated from the domain *o=AC-based domain,c=C*. There is only one allowed action, *translate*. The example defines that the attribute type *studentRole* with value *ERASMUS* must be translated into the internal attribute type *urn:role:student*, with value *ERASMUS*.

5. Policy enforcement

In a XACML scenario, when a subject needs to execute an action on a particular resource, he sends a query to the entity protecting the resource, the Policy Enforcement Point (PEP). The PEP builds a request, using a native format, based on the subject's attributes, action, resource, and other relevant information. Then, it sends this request to a Context Handler element, which acts as a bridge between the PEP and the Policy Decision Point (PDP), and which translates the native request into a XACML Request object. The Context Hander sends the request to the PDP, which, according to the access control policy, gives an authorization decision determining whether the access should be granted. That answer is returned to the Context Handler, as a XACML Response object, which responds to the PEP in native format.

According to the NAS-SAML scenario, the AAA acts as the PEP, and the PDP located in the target domain is in charge of giving the authorization decisions. A new module has to be added to the AAA in order to implement the Context Handler functionality. Communications between the PEP and the Context Handler are made through the SAML request-response protocol.

Following the push and pull models described in the network access scenario, this section describes how the involved elements interact during the enforcement process, that is, when the *Resource Access Policy* is used to grant or deny the access to the desired resource.

5.1. Obtaining the user attributes

In the pull model, when the AAA server needs to obtain the user attributes it has to locate first the Source Authority (SA) which controls this assignment. This SA can be located in the same domain (single-domain model) or in a foreign one (inter-domain model). In this way, the AAA server sends an attribute query to the SA, indicating the user subject and the demanded resource. The SA checks the *Role Assignment Policy*, selects the attributes to be disclosed and responds to the AAA server with the selected user's attributes. The request and response messages exchanged in this stage between the AAA server and the user's SA are defined as SAML Request and Response sentences, and transported over the AAA network infrastructure. If the AAA server detects the user belongs to a foreign domain based on different authorization credentials, it sends the request to a conversion service [8]. This service uses the *Conversion Policy* to translate them. Once translated, the attributes are returned to the AAA server.

In a push model, the user directly presents his attributes to the AAA server during the network access attempt.

5.2. Request access control

Once the AAA server has the user's attributes it sends an authorization decision query to the Context Handler module, which will be in charge of sending the XACML request and obtaining the XACML response from the PDP. First, the Context Handler sends an XACML Request asking whether the home SA is recognized. Then, the PDP obtains the *Role Allocation Policy* and returns a XCAML Response message with a decision. If the Context Handler receives a positive response it sends another XACML Request message asking whether the user, according to the role(s) played, has permission to access to the desired resource and to execute the specified action. The PDP then enforces the *Target Access Policy* and returns a XACML Response message containing the decision and the obligations derived from that decision. The Context Handler returns the decision and the set of obligations, if positive, to the AAA server, which will be in charge of enforcing them.

6. Related work

This section describes two solutions that define access control scenarios and their related policies, Shibboleth [4] and the PERMIS project [2]. Both of them define policies using its own XML-based policy syntax. We briefly describe how these systems work and whether the policies proposed in this paper are suitable for them.

Shibboleth defines an access control approach scenario for web environments, which is composed of three main entities: service providers offering target resources; identity providers maintaining the user's identities; and end users. It offers user authentication and attribute-based authorization services based on SAML, as well as a SSO service. It is based on a single Attribute Authority.

The Attribute Authority maintains a set of policies called *Attribute Release Policies* (or ARP's) that govern the sharing of user attributes with Shibboleth target sites. When a user attempts to access a Shibboleth-protected resource, that resource's obtains only the attributes/values allowed in this policy. On the other hand, when a target site receives a set of user's attributes, it is configured to accept specific attributes that it understand. Target site evaluates the attributes and values to grant or deny access to the desired resource. The set of rules defining these decisions is called an *Attribute Acceptance Policy* (AAP).

Although Shibboleth defines its own XML policy syntax to represent ARPs, XACML can be used, as described in [11], to represent ARPs and AAPs policies. In fact, the set of policies described in this paper could be adapted, in order to meet scenario requirements, to defined the Shibboleth policies. ARP can be represented by means of the *Role Assignment Policy*, adding disclosure rules for the target sites, as described in [8]. AAP rules can be described by means of the *Role Allocation Policy* and *Target Access Policy*. Moreover, *Conversion Policy* adds a way to extend the Shibboleth domains with other domains based on different authorization schemas.

PERMIS [2] is a trust management system based on X.509 Attribute Certificates to specify subject attributes such as roles and permissions. It also defines a hierarchical role based access control (RBAC) policy language in terms of those roles and permissions. The PERMIS policy specifies who is to be granted what type of action on which targets, and under what conditions. Policy based authorization on the other hand allows the domain administrator (the SOA) to specify the policy for the whole domain, and all targets will then be controlled by the same set of rules. The policy is expressed in XML and comprises the following components:

- *SubjectPolicy*: defines users from a subject domain that may be authorized.
- *RoleHierarchyPolicy*: defines roles and their relationships to each other.
- *SOAPolicy*: defines SOAs trusted to allocate roles.
- *RoleAssignmentPolicy*: defines which roles may be allocated to which subjects by which SOAs.
- *TargetPolicy*: defines the target domains covered by this policy.
- *ActionPolicy*: defines the actions (or methods) supported by the targets.
- *TargetAccessPolicy*: defines which roles have permission to perform which actions on which targets, and under which conditions.

The SIPS project [12] has integrated Shibboleth and PERMIS [13]. The proposed set of policies can be easily adapted to be used in a PERMIS scenario since the *Role Allocation Policy* and *Target Access Policy* can be used to represent this PERMIS policy.

7. Conclusions

This paper identifies the set of policies required in a network access control environment as the one proposed in [1]. We have described policies which involve the whole user's attributes life cycle, from the assignment process in the user's home domain to the resource access control process, in the target domain, when the user makes use of his attributes.

The policies presented have been described through XACML, which has been designed to be a standard way to express access control policies. The use of XACML and the definition of policies compatible with other important access control scenario helps in the definition of interoperable access control policies and the definition of heterogeneous environments.

We have also defined a scenario where we can show how XACML and SAML can be easily integrated and how they can work together in the definition of authorization attributes, the transport of them between the entities involved and their use in access control decisions. Furthermore, we can see how other non-XML authorization models can be integrated by means of conversion services.

As a statement of direction, we are working in the definition of the proposed scenario for high level application solutions, as GRID environments, and the policies related to this kind of systems.

Acknowledgements

Partially supported by IST-2002-001929 (SEINIT)

References

[1] G. López, O. Cánovas, A. F. Gómez, and R. Marín. A Network Access Control Approach based on the AAA Architecture and Authorization Attributes. In *Proceedings of International Workshop on Systems and Security Networks (SSN05)*, April 2005.
[2] David W. Chadwick, Alexander Otenko, and Ed Ball. Role-based access control with x.509 attribute certificates. *IEEE Internet Computing*, 7(2):62–69, 2003.
[3] *Akenti Distributed Access Control*, April 2005. http://www-itg.lbl.gov/Akenti/.
[4] S. Cantor. *Shibboleth Architecture. Protocols and Profiles*, February 2005. http://shibboleth.internet2.edu/docs/draft-mace-shibboleth-arch-protocols-06.pdf.
[5] S. Godik and T. Moses. *OASIS eXtensible Access Control Markup Language (XACML) Version 2.0*, February 2005. OASIS Standard.
[6] LAN MAN Standards Committee of the IEEE Computer Society. *IEEE Draft P802.1X/D11: Standard for Port based Network Access Control*. IEEE, March 2001.
[7] C. de Laat, G. Gross, L. Gommans, J. Vollbrecht, and D. Spence. *Generic AAA Architecture*. Internet Engineering Task Force, August 2000. Request for Comments (RFC) 2903.
[8] G. López, O. Cánovas, A. F. Gómez, and R. Marín. A heterogeneous network access service based on permis and saml. In *Proceedings of 2st European PKI Workshop*, June 2005. To be published.
[9] S. Farrel and R. Housley. *An Internet Attribute Certificate Profile for Authorization*. Internet Engineering Task Force, April 2002. Request for Comments (RFC) 3281.
[10] Anne Anderson. *Core and Hierarchical Role Based Access Control (RBAC) profile of XACML, Version 2.0*, September 2004. OASIS Committe Draft 01.
[11] M. Lorch, S. Proctor, R. Lepro, D. Kafura, and S. Shah. First experiences using xacml for access control in distributed systems. In *ACM Workshop on XML Security*, 2003.
[12] SIPS. *Seamlessly Integrating PERMIS and Shibboleth*, February 2005. http://www.jisc.ac.uk.
[13] David Chadwick, Sassa Otenko, Wensheng Xu, and Zhi Qing WuK. Adding distributed trust management to shibboleth. In *Proceedings of the 4th Annual PKI R-D Workshop*, April 2005.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

A Delegation Logic Based Authorization Mechanism for Virtual Organizations

Chunhua GU, Xueqin ZHANG and Guoxin SONG
College of Information Science and Engineering in East China University of Science and Technology, Shanghai, China

Abstract. Because of the new business trends such as cooperating, downsizing and resource sharing, the use of virtual organization (VO) is gaining increasing importance as a model for building large-scale business information systems. Authorization is essential in VO in order to control the access to shared resources. But authorization in VO is challenging because the participants of VO need to collaborate in a distributed, dynamic and heterogeneous environment, and accordingly the access control policies are complex. A delegation logic based authorization mechanism is put forward in this paper. Our proposed approach translates the access requests, credentials and access policies into unified delegation logic rules. Based on the calculation on those rules, the access decision is made. We introduce the concept of Access Unit (AU), which wraps the AC system of a task. The rule exchange interface of AU is defined. The main contribution of this paper is that it suggests a practical mechanism for implementing authorization for VO. In essence, we propose an approach to enforce RBAC in VO based on task/project structure.

Keywords. RBAC, Distributed systems, PKI, Virtual Organization

1. Introduction

With the rapid development of the Internet, the design of distributed computing systems is greatly impacted. Today distributed computing systems typically cover wide geographic area and are used as a model for organizing and implementing large-scale business application. A new type of business built from both organizationally and geographically distributed, virtual organization[1], is being attractive because of business trends such as cooperating, downsizing and resource sharing. VO is an abstraction for designing distributed systems that aims to provide a flexible abstraction for implementing distributed systems with complex inter-process relationships. The operation of a VO involves, firstly, the enrolment of participants and their resources and, secondly, the establishment of relationship among participants and resources within the VO in order to collaborate and to achieve the specific task. Thus the implementation and management of VO require that these two operations be carefully controlled and in accordance with the security policies of the VO. However, the tasks of specifying and enforcing the policies are challenging in that each participating organization may nominate a large number of users to access the shared resources and, for accountability and liability reasons typical in business applications, controlled access authorized by a group of users is required. Furthermore, in general, VO participants join and leave the VO in a dynamic manner. A key requirement of VO is the ability to negotiate resource-sharing arrangement among a set of member organizations and then use the resource pool for achieving some application objectives[2]. The connection of the member organizations in VO is temporary and dynamic. The cooperative organizations may be opponents sometime in future. Therefore the control of accessing resources in VO is very important and complicated.

Several access control models have been introduced such as DAC（Discretionary Access Control）[3], MAC（Mandatory Access Control）[4], and RBAC（Role Based Access Control）[5][6]. But these traditional AC models are only suitable for the single real organization. To meet the different situations and increase the flexibility and capability, many extended AC models have been proposed. Some AC systems[7][8][9][10][11] have been developed for collaborative environment such as workflow systems. Task-based Authorization Controls (TBAC) [12][13] is a task-oriented model for access control and authorization. In the TBAC paradigm, permissions are revoked and granted just-in-time fashion based on activities and tasks. The SALSA security architecture[14] is another extension from centralized AC models. It consists of two AC modules: (a) an organization-specific access control module (OACM) which is controlled by each organization and enforces the security policies set by each organization; (b) a task-specific access module (TACM) which is controlled by each workflow and enforces task-specific security policies. Because the task-specific security policy depends on the semantics of the workflow, only the person with intimate knowledge of the workflow can set the security policies for each task. SecureFlow[15] associates each task in the workflow with Authorization Templates (AT), which is specified by the workflow designer. According to ATs, authorization is actually granted when the task is started. Layer Based Access Control (LBAC) [16] puts forward a way to exchange access control information dependent on process decomposition and task assignment. The inter-organization access control is implemented by updating the local ACL.

Based on the previous work described above, we propose a delegation logic based authorization mechanism which protects objects in different levels, supports the dynamic allied environment, and provides a uniform way to handle both internal and inter-organizational access control. We provide a formal way to express and transmit the credentials and security rules through which the access decision is made. This paper is organized as follows, we address the authorization requirements in virtual organization and analyze the existing AC models in section 2. In section 3 we briefly introduce the notion of delegation logic rule and logic-programming language D1LP. The project/task structure and extension on RBAC are introduced in section 4. In section 5, the delegation logic based authorization mechanism is described. Section 6 explains how it works with inter-organizational access control through examples. Finally we give our conclusion in section 7.

2. Authorization in Virtual Organizations

The main characteristics that distinguish the authorization systems in virtual organization from general authorization systems are task assignments, dynamic associations, and distributed locations. Not only does it provide the way to control the information sharing in VO, but also it protects the information independence in a real organization. Each real organization has its own AC system. The AC system of a virtual organization is a complicated system in which the AC systems of its member organizations are included. Operations on a VO typically involves the following:

- Registration of participants and their resources to form the VO.
- Registration of shared resources by these participants so as to allow them to collaborate in order to achieve some specific tasks.
- Nomination of users authorized to access the resources on behalf of the participants.
- Specification of policies for controlling access to shared resources.
- Enforcement of access control policies when users access resources during the course their work.

Authorization in virtual organization is a challenging task due to the complicated characteristics of VO. It needs to support the following functions:

a) Protect objects between real organizations. There are two kinds of subjects:(1) those who are in the same real organization with the objects, (2) those who belong to the other real organization. The protection policies should be different because the trustworthiness of those subjects is different.
b) Support dynamic relations. The members of VO keep on changing dynamically. The AC system should provide a consistent way to handle the latest inter-organizational protection within VO.
c) Support heterogeneous collaborative environments. Each real organization has its own AC policy and AC system in its own network environment. The AC system of a virtual organization should support different collaborative environments.
d) Support different collaboration modes. The member organizations in VO have different collaboration modes which lead to different structures such as linear structure, tree structure and net structure. Authorization system in VO should support different structures.

In a distributed system, it is necessary to define a standard method through which the original authorization systems in real organization can exchange AC information to form the authorization system of virtual organization. We need to develop a framework that defines conceptual boundaries around projects and tasks so that the roles and permissions can be scoped. Since a VO is comprised from a set of services, messages between those services need to propagate context related information that can be used to identify the specific scope in which the security-related decisions can be made. Services can use the context information to determine whether the requestor has sufficient permission to perform operations or access resources. In this paper, a delegation logic based Authorization mechanism is put forward for virtual organization environment.

3. Delegation Logic Rule

Blaze[17] defines trust management as “a unified approach to specifying and interpreting security policies, credentials, relationships which allows direct authorization of security of security-critical actions”. In the “trust-management” approach to distributed authorization, a “requester” submits a request, possibly supported by a set of “credentials” issued by other parties, to an “authorizer,” who controls the requested resources. The authorizer then decides whether to authorize this request by answering the “proof-of-compliance” question: “Do these credentials prove that a request complies with my local policy?”

Ninghui[18] extends well understood logic-programming languages with features needed for distributed authorization and provides a logic-based approach to distributed authorization -- Delegation Logic(DL). DL extends Definite Ordinary Logic Programs along two dimensions: delegation and nonmonotonic reasoning. DL rules are used to represent access control policies and credentials. In this paper, we use a monotonic Delegation Logic that is called D1LP to present delegation logic rules. In the following, we only list some of D1LP syntax elements. For the details of D1LP, the reader is referred to the paper[18].

1. A term is either a constant or a variable, e.g. Bob, ?X . When a variable appears in certain positions, it is called a principal variable and can be instantiated only to a principal. A principal term is either a principal or a principal variable.
2. A base atom encodes a belief. For example, the base atom “remove(file1)” encodes the belief that file1 should be removed.
3. In a direct statement “X says ba,” X is called the issuer of this statement. This statement intuitively means that X “supports” the belief encoded in ba. For example, a direct statement “Bob says remove(file1)” means that Bob supports that file1 should be removed; i.e., it represents Bob’s request to remove file1.
4. In a delegation statement “X delegates ba^d to PE,” X is called the issuer of this statement; d is called the delegation depth of this delegation; and the principal expression PE is called the delegatee of this delegation. The delegation statement “Bob delegates goodCredit(?X)^1 to Carl” means that, if Carl supports that someone has good credit, then Bob supports it as well.
5. A dynamic unweighted threshold structure “threshold (k, ?X, Prin says ba)),” in which we require that ?X appears in ba and define the threshold pool to be the set of all principals A such that the direct statement “Prin says pred(. . .A. . .)” is true, i.e., the expression “Prin says pred(. . .?X . . .)” becomes true when A is substituted for ?X for each appearance throughout the direct statement.
6. A rule is also known as a clause. In a rule “H if F,” H is a direct statement, a delegation statement, or a representation statement and is called the head of the rule; F is a formula of body statements and called the body of the rule. The body may be empty; if it is, the keyword “if” is omitted. A rule with an empty body is also called a fact. A rule with an empty body is also called a fact. If a rule’s head H has a principal as its issuer, then this principal is also the issuer of this rule. Otherwise H has a principal variable as its issuer, and the issuer of this rule is the principal symbol Local.
7. A program is a finite set of rules plus an optional locale-declaration statement. The locale-declaration statement is required if Local appears in any of the rules. The set of rules is also known as a logic program (LP) or as a rule-set.
8. A query takes the form “F?” where F is a body formula.

We use delegation logic rules for representing the credentials and access control policies in virtual organization. Entities in a distributed system are expressed by principals in DL rules. These principals issue credentials and requests. Typically, a principal in distributed AC system is a public/private key pair. Such a principal issues a credential or a request by digitally signing a message that contains it.

When a subject from real organization E1 needs to access the object that is in another real organization E2, the related credentials and request are sent to E2. After E2 gets a request and some credentials that support this request, it creates a query Q from this request and a DL program (rule-set) P from the combination of the credentials and its local policies. Policies and credentials are translated into rules in DL. During the translation, E2 should verify that each rule is indeed made by its issuer. A rule with a principal as its issuer should be encoded in a credential that is signed by the rule's issuer. A rule with Local as its issuer should come from a local policy. Policies that are securely stored locally do not need to be signed. Having a program P and a query Q, E2 decides whether to authorize this request by inferring whether the query Q is true relative to the program P. DL's semantics gives a proof procedure to answer Q relative to P.

4. Project-Task Structure in VO

A VO is composed of participants from different organizations driven by specific projects[13]. A project can be divided into some tasks. In order to achieve the level of granularity we need to provide several levels of access control enforcement. In the first instance these manifest themselves as boundaries; that is project and task boundaries. The task boundary encapsulates all roles and objects related to an atomic action or activity. A project boundary encapsulates all tasks – atomic and otherwise – that take place within a VO in order to achieve a common goal. The concept of task is an implementation of the concept of spheres of control and is employed to provide a conceptual boundary encapsulating a specific action. Within its boundaries we find the following:

- A task name, i.e. unique identifier of a particular instance of a task;
- A set of roles;
- Objects/resources used by that task;
- Reference(s) to resource(s) relevant to the task outside the tasks boundary;
- Pre-conditions, Post-conditions.

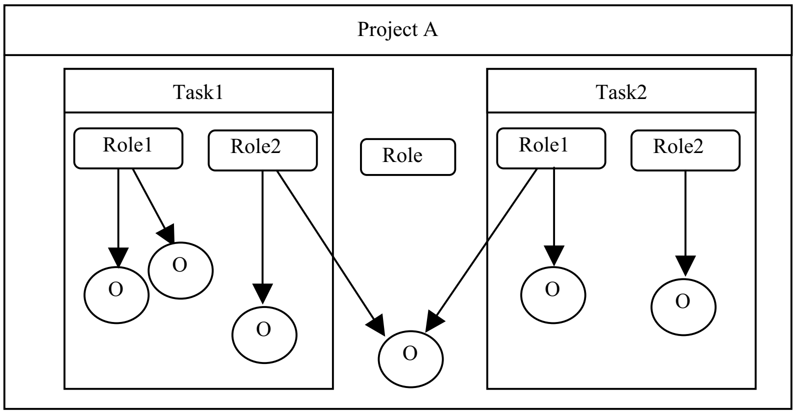


Figure 1. Project/Task structure

Figure 1 provides a conceptual representation of tasks in a project. The outer box represents the project. The inner box denotes the boundaries of the task within which the instances of roles and the set of access permissions are active. This logical task can span across multiple organizations. Each of the resources/objects in Figure 1 is associated with access permissions. The system evaluates access to the resources according to the role that attempts to perform an action always within the context of a particular task. The Figure 1 also illustrates the relationship between projects, tasks, role instances, and the objects/resources. The project would be a VO that is established for the development and market exploitation of a chemical compound while a task could be a chemical hazard analysis. Objects that are shared between tasks (as project-bound objects) can be VO-wide documents while task-bound objects can be temporary notes used for the completion of that task. Roles may evolve according to the contracts put in place to manage the VO. For example, an employee from one company with an assigned role L may have access to some of the documents from another company during task B but will be denied access to them during the preceding task A. The contracts that are in place to establish a VO determine the roles and their permissions for each of the task that VO is to perform. The runtime support for the VO monitors the progress of each task in relation to the contracts in place and dynamically changes the permissions of the roles where appropriate.

In VO, the tasks are assigned between the organizations. Task becomes a critical factor when controlling the access to resources. Therefore we extend RBAC by considering task as a part of AC. The extended RBAC has following components:

a) U, R, T, P and S，(users, roles, tasks, permissions, and sessions respectively);
b) $PA \subseteq P \times R$, a many-to-many permission to role assignment relation;
c) $UA \subseteq U \times R$, a many-to-many user to role assignment relation;
d) $PT \subseteq P \times T$, a many-to-many permission to task assignment relation;
e) *user*: S→U，a function mapping each session s_i to the single user *user*(s_i);
f) *roles*:S × →2^R，a function mapping session s_i to a set of roles:
g) $roles(s_i) \subseteq \{r \mid (user(s_i), r) \in UA\}$, *Session* s_i has the permission:
h) $\bigcup_{r \in roles(s_i)} \{p \mid (p, r) \in PA\}$.
i) *perms*:$R \times T$→2^P，a function mapping role r_i and task t_i to a set of permissions:
j) $perms(r_i, t_i) \subseteq \{p \mid ((p, r_i) \in PA) \wedge ((p, t_i) \in PT)\}$.

5. Delegation Logic Based Authorization Mechanism

In this section, we present a delegation logic based authorization mechanism which supports distributed computing system for virtual organization. On the base of RBAC, our proposed approach protects objects between real organizations in different level of granularity. It allows a real organization to have its own specific access control environment. The authorization system of each member organization provides interfaces through which the original authorization system can exchange information with outside.

5.1. Functionalities

5.1.1. Access Control between Member Organizations

RBAC is adopted to handle access control inside a member organization of VO. Roles are task-oriented and defined within an organization. The RBAC elements are translated into DL rules and then associated with the DL rules from other organizations. From those DL rules, an access control decision is inferred. In an organization, a DL rule library is maintained to control access from outside.

VO has a Source of Authority (SoA) which is the root of trust in VO. Some credentials are sent to SoA before a real organization joins the virtual organization. For example, company A is a member of virtual organization. It wants to outsource task T1 which is part of project P to company B. The following credentials are represented by D1LP and sent to SoA:

A says isproject(P).
A says allowenter(B, T1).
A says ischildtask(T1, T). //if T1 is a task of project P

5.1.2. Basic Policies for Permission Setting

When a task is assigned between real organizations, it contains two parts: a task template (containing information for the assignee to complete the task) and contracts (exchanging AC information). Along with the task between real organizations there are two contracts, a delegation report and a local contract. The delegation report defines the DL rules and what object information should be returned from the assignee to the assigner. This information will be parsed and explained in assignee side. The local contract defines who can access the objects information returned by the assignee. The local contract is explained by the AC system in assigner side. The task template and the report contract are sent to the assignee organization, and the local contract is kept in the assigner's system.

From the viewpoint of AC, the objects information related to the task belong to assignee's organization when a task is assigned across the boundary of organizations. For security reason, these objects should not be directly accessed from outside of the organization. A 'delegation report' is designed to solve this problem. The access request and DL rules are encapsulated into the delegation report. According to the delegation report, the assignee's AC system infers the decision from access policies and DL rules, and then returns the requested objects back to the assigner organization. Based on the local contract, the assigner AC system sets permissions for each object reported by assignee, and updates the ACL. Assignee's ACL is updated according to delegation report and object owner.

5.1.3. AC Information Exchange

There are two types of interfaces in an AU. One is used to receive AC information, called contract, the other is used to transmit AC information, called report. Through these interfaces, (1) AC and object information is exchanged; (2) the AC system detail in an organization is hidden; (3) the original AC systems are extended to let them talk with each other in a standard way; (4) the DL based "proof-of-compliance" is provided to handle AC problems; (5) the inter-organizational AC is converted into internal AC.

5.2. Authorization Mechanism

A virtual organization dynamically combines different real organizations into a collaborative environment. The major consideration of designing authorization mechanism for VO is to allow different authorization systems to freely exchange DL rules and credentials with one another so that they can be integrated easily. To achieve this goal, the Object-Oriented approach is used to support several OO features: abstraction, encapsulation, messages and dynamic binding. An entire AC system is treated as an object. To differentiate with the concept "object" used inside the AC system, the entire AC system, including all subjects, objects and actions, is called as an access unit (AU). An AU, which wraps the whole AC system inside, is an abstract of an entire AC system. All AC information is hidden in the AU. The information is exchanged by using messages through the interface of the AU. An interface contains two parts: a contract and a report. The contract is used to receive DL rules and credentials from outside, and the report is used to send task-related object information to outside. The interface is designed for an AU to allow the functions inside the AU to exchange messages with outside. Because the interface is used, the AU supports dynamic binding, which occurs when a task is assigned or a real organization joins the virtual organization dynamically. The basic AU is shown in Figure 2.

An AC system of a real organization is an AU. The AU of a virtual organization is composed of many AUs . Consider the virtual organization with layered structure, the architecture of its AC system is shown in Figure 3. The steps of how the information is exchanged between the adjacent AUs. (1) Layer0 receives DL rules and AC information through its contract from outside higher AU; (2) DL parser of layer0 explains and calculates the DL rules, credentials and local rules, (3) the result information is merged into its own AC system. When a task is assigned from layer0 to layer1, (4) the AC system in layer0 creates a contract including related credentials and DL rules, and sends the contract to layer1, and AC information in the contract of layer1 is also added into layer1's AC system, so that the AC information, including DL rules and credentials, is sent from a higher AU to a lower AU following the task assignment. If the AC system of a lower AU permits the access request from the higher AU, certain objects are contained in its report, and these objects can be accessed from outside. (5) The objects in a report of a lower AU are added into the object set of the higher AU. As illustrated in Figure 3, the object set in layer0 contains the objects in the reports of layer1. These objects could be part of the report of layer0. Then go through (6) and (7) object information (data or process) is sent to higher layers.

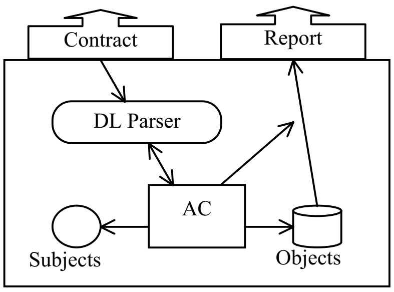


Figure 2. Basic Structure of AU

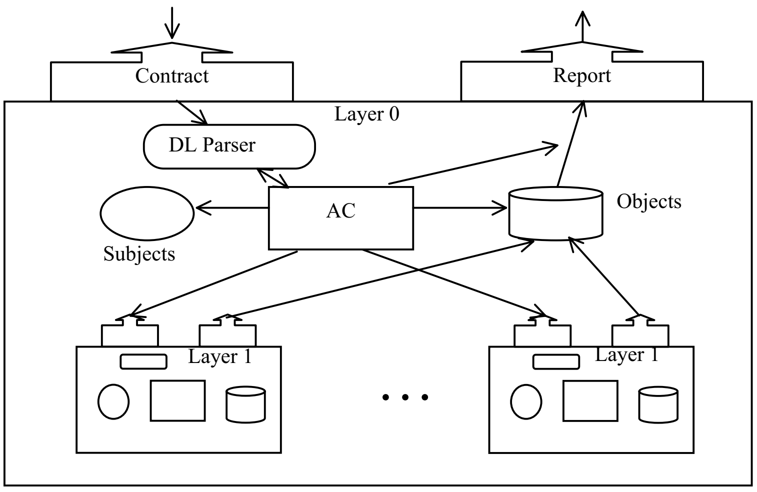


Figure 3. Authorization mechanism

Another feature is that the higher AU always hides information of the lower AUs inside it. So the entire AC system for a virtual organization is like a tree, the higher AU encapsulates the lower AUs. By using this authorization mechanism the difficulties of the AC system for a collaborative environment are solved.

5.3. Conversion from RBAC to DL Rule

While the received credentials and access requests are converted into DL rules, the RBAC elements (including users, objects, tasks, permissions, user-role mapping and role-permission mapping) are converted into DL rules too. The conversion rules are shown in Table 1.

Table 1. The Conversion Rules

Notions in RBAC	Notions in Delegation Logic
Users	Principals
Objects	Resource Constants
Roles	Assertions, e.g. isRole()
Tasks	Constants
User-role mapping	Fact Rule, e.g. A says isManager(Bob)
Permissions	Operation (BA)
Role-permission mapping	Rule, e.g. A says BA(?X) if A says isRole(?X)
Task-permission mapping	Fact Rule, e.g. A says isRelated(T, O)

5.4. DL Rule Representation

All authorization related DL rules, such as credentials, access rules, and access request are represented by using XML, which makes this data easy to understand and managed. From this point of view, our authorization system is an XML based system. Similar to our purpose, IBM introduces XACL (XML Access Control Language), which enables AC system designer to specify and execute fine-grained and complex authorization policies. Comparing with XACL, the main purpose of using XML is different. Our approach uses XML to record authorization information.

5.4.1. The XML Representation of Credential

```
<credential op="Delegates">
        <sissuer>Abc</sissuer>
        <ba baname="isABC">
                <param>Abc</param>
        </ba>
        <delegateobject>
                <depth>2</depth>
                <complexobject>
                        <threshold>
                                <k>1</k>
                                <credential op="Says">
                                        <sissuer>Abc</sissuer>
                                        <ba baname="isABC">
                                                <param>?x</param>
                                        </ba>
                                </credential>
                        </threshold>
                </complexobject>
        </delegateobject>
</credential>
```

5.4.2. The XML Representation of Access Rule

```
<rule>
        <header>
                <credential op="Says">
                        <simpleissuer>Abc</simpleissuer>
                        <ba baname="Abc">
                                <param>Abc</param>
                        </ba >
                </credential>
        </header>
        <body>
                <credential op="Delegates">
                ……
                </credential>
        </body>
</rule>
```

5.4.3. The XML Representation of Access Request:

```
    <request>
        <credential op="Says">
                    <simpleissuer>Abc</simpleissuer>
                    <ba baname="isABC">
                                <param>Abc</param>
                    </ba>
         </credential>
    </request>
```

6. An Example

In this example, we illustrate how a project is handled by collaboration of different companies and how access is controlled. Suppose that company *A* plans to design a new engine, and starts a new project called "engine design". The engine design process is decomposed into different subtasks. We assume that some subtasks are handled by other engineering companies that construct a virtual organization environment with company *A*. The company *A* decides to outsource a part of the process. Company *B* is a professional cylinder design company. Company *A* asks company *B* to design the cylinder part for the new engine (the grey part of Figure 4).

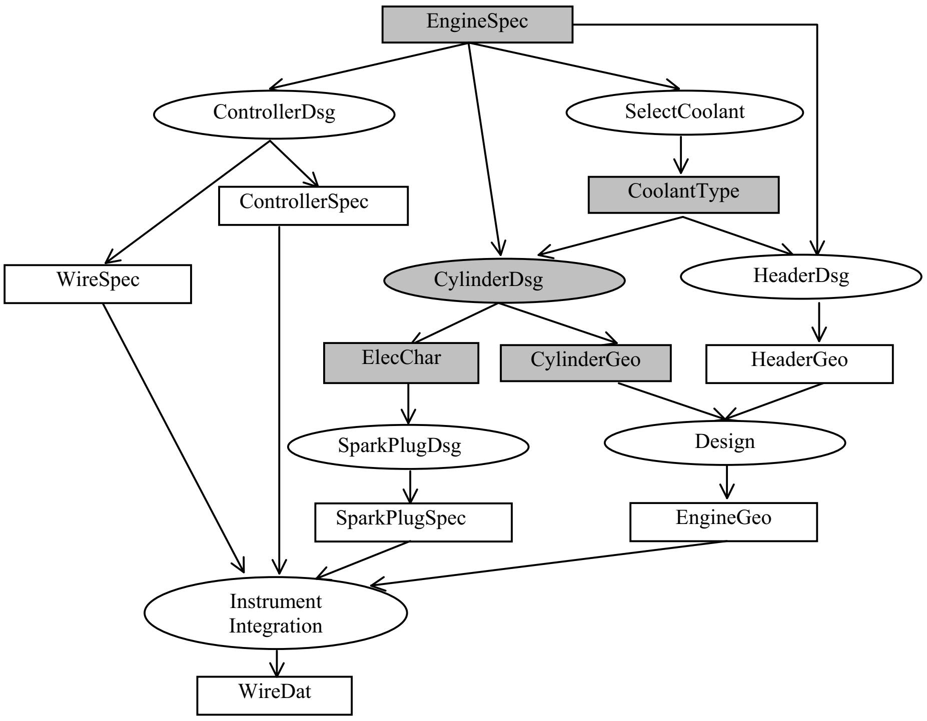


Figure 4. The process flow of "engine design"

VOCA is the SoA trusted by all member organizations. Each real organization owns its private key and gets certificate from VOCA. Before the data are transmitted on Internet, they must be encrypted or signed. Company *A* and its collaborative partners have their independent RBAC systems.

The manager in company *A* selects the logical task 'CylinderDsg' and generates the task 'Cylinder Design' for company *B*. The task contains two parts: task template and contracts. There are two contracts in this task: delegation report and local contract. The task template and delegation report are sent to company *B*'s system and local contract is kept in the system of company *A*. After receiving the task, the AC system in company *B* parses the delegation report and starts the task. During the task processing, the manager of company *A* can trace the task execution in company *B*. When the task is finished, the requested objects are put into report interface of company *B* and returned back to company *A* according to the delegation report. After the assigner receives the report, the assigner's AC system automatically creates ACLs for those objects in the report and set permissions based on the local contract.

"Cylinder Design" is a subtask. There are data object 'EngineSpec' and 'CoolantType' in task template. The delegation report contains data objects 'ElecChar' and 'CylinderGeo', which are requested to return. Some DL rules and launch rules are also contained in delegation report. The permissions to objects 'ElecChar' and 'CylinderGeo' are put in local contract. After receiving the task and AC information from the contract interface, a manager of company *B*, Bob, wants to launch this task. A query "Does company *A* allow Bob to launch this task?" is formed. It can be described in D1LP:

comA says canLaunch(taskCD,Bob)?

The launch rule in delegation report is "The manager of collaborative company can launch this task". The rules from delegation report are:

comA says canlaunch(?T,?X) if comA says isManager(?X).
comA delegates isManager(?X) to threshold(1,?C,comA says
isPartner(?C,comA, taskCD)).
comA delegates isPartner(?C,comA, taskCD) to VOCA.

These rules are from company *A*. The first one is a launch rule. The other rules are delegation logic rule.

The related credentials existed in VOCA and company *B* are:

VOCA says isPartner(comB,comA, taskCD).
comB says isManager(Bob).

The DL parser in company B explains above rules and gets the conclusion: Bob can launch the task "Cylinder Design".

Assuming both Alice and Tom are employees of company *A*. Alice is a manager who is responsible for task "Cylinder Design". During the task execution, Alice and Tom want to know the status of the task, i.e. access the data object 'ElecChar'. Two delegation reports are sent to the contract interface. The access requests and related credentials are included in those delegation reports. The requests can be represented in D1LP queries as:

comB says viewObj(ElecChar,Alice) ?
comB says viewObj(ElecChar,Tom) ?

In company *B*, the rule of allowing access from outside is: "the task related project manager can access the information of that task". It can be described in D1LP:

comB says viewObj(?subT,?X) if
comB says isProjManager(?X,?T),comB says ischildtask(?subT,?T).

comB delegates isProjManager(?X,?T) to
threshold(1,?C,comB says owner(?T,?C).

Other rules included in delegation report:

comA says isStaff(Alice).
comA says isStaff(Tom).
comA says isProjManager(Alice,CylinderDsg).

And credentials in company B：

comB says owner(CylinderDsg,comA).
comB says ischildtask(ElecChar,CylinderDsg).

From the above rules, company B can get the positive answer to "comB says viewObj(ElecChar,Alice) ?" and the negative answer to "comB says viewObj(ElecChar,Tom)". The conclusion is: Alice is allowed to access while Tom is denied.

7. Conclusion

In this paper, we addressed the special authorization requirements raised in virtual organization environment. Our proposed approach merges the ideas of RBAC and trust management, which provides a formal way to express and transmit the credentials and security rules. It extends RBAC to inter-organizational environment driven by task assignment and supports authorization information exchange among different access units in a virtual organization. We have shown that it supports authorization in heterogeneous environments. Currently, a prototype is implemented using JAVA to make the system platform-independent.

References

[1] I. Foster, C. Kesselman, S. Tuecke. The anatomy of the grid: enabling scalable virtual organizations, Int. J. Supercomputer Applications, 2001, 15(3), 200-222.
[2] Germano C, Sandeep K, Christoph S. et al.. Virtual Enterprise Networks: The Next Enterprise Networking. In Proc. of the Sixteenth Annual Computer Security Applications Conference 2000. New Orleans, Louisiana: IEEE, 2000, 42-52.
[3] R.S. Sandhu and P. Samarati. Access control: Principal and practice. IEEE Communications Magazine, 1994, 40-48.
[4] D.E. Bell and L.J. LaPadula. Secure computer systems: Mathematical foundations and model. TR. M., 1973, 74-224.
[5] Gregory S, Michael H, Vijay V. Role-Based Access Control and the Access Control Matrix. In Proc. of ICICS 2003. Springer, German, LNCS 2836, 2003, 145-157.
[6] R.S. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman. Role-Based Access Models. IEEE Computer. 1996, 29(2), 38-47.
[7] V. Atluri and W. Huang. An authorization model for workflows. In Proc. of the 4th European Symposium on Research in Computer Security, Rome, Italy, 1996, 25-27.
[8] A. Bullock and S. Benford. An access control framework for multi-user collaborative environments. In Proc. of the international ACM SIGGROUP conference on Supporting group work, Phoenix, Arizona, USA, 1999, 140-149.
[9] HyungHyo L, SeungYong L, BongNam N. A New Role-Based Authorization Model in a Corparate Workflow Systems. In Proc. of ICCSA 2004. Springer, German, LNCS 3043, 2004, 701-710.
[10] William B. An Architecture for the Virtual Enterprise. In Proc. of the IEEE International Conference on System, Man and Cybernetics. Piscataway, NJ, USA, IEEE, 1994, 94CH3571-5.
[11] P. Linington, Z. Milosevic, and K. Raymond. Policies in communities: Extending the odp enterprise viewpoint. In Proc. 2nd IEEE Enterprise Distributed Object Computing Workshop, San-Diego, 1998.

[12] R.K. Thomas and R. S. Sandhu. Task-based authorization controls (tbac): A family of models for active and enterprise-oriented authorization management. In the IFIO WG11.3 Workshop on Database Security, Lake Tahoe, California, 1997.
[13] N. Guelfi et al, Task-Based Access Control for Virtual Organizations. In Proc. of FIDJI 2004, LNCS 3409, Springer, German, 2005, 38-47
[14] M.H. Kang, J. S. Park, and et al. Access control mechanisms for interorganizational workflow. In In Sixth ACM Symposium on Access Control Models and Technologies, 2001.
[15] W. Huang and V. Atluri. Secureflow: A secure web-enabled workflow management system. ACM Workshop on Role-based Access Control, 1999, 83–94.
[16] Yuan Zhang, Moon Jung Chung, and Hyun Kim. Layer-Based Access Control Model in the Manufacturing Infrastructure and Design Automation System. In Proc. of ICISC 2003, Springer, German, 2004, 197-214
[17] Matt Blaze, Joan Feigenbaum, John Ioannidis, and Angelos D. Keromytis. The KeyNote trust-management system, version 2. IETF RFC 2704, 1999.
[18] Ninghui Li, Benjamin N. Grosof, and Joan Feigenbaum. Delegation Logic: A Logic-based Approach to Distributed Authorization. ACM Transactions on Information and System Security (TISSEC), 2003, 6(1), 128-171.

Authentication & Time-Stamping

This page intentionally left blank

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005
© 2005 The authors. All rights reserved.

An Efficient and Flexible Scheme to Support Biometric-Based and Role-Based Access Control

Deholo Nali [a,1], Carlisle Adams [a] and Ali Miri [a]
[a] *School of Information Technology and Engineering (SITE), University of Ottawa*

Abstract. Introduced at EuroCrypt'05, threshold attribute-based encryption (thABE) is a subclass of identity-based encryption which views each identity as a set of descriptive attributes. In order to decrypt a ciphertext c encrypted for a set ω of attributes, users must have attribute keys associated with a sufficiently large subset of ω. Applications of thABE include both biometric-based and role-based *cryptographic access control*. This paper presents an efficient and flexible thABE scheme which is provably secure in the random oracle model. Let d be a minimal number of attributes which a decryptor must have to decipher a ciphertext. The proposed scheme requires only two pairings for decryption (instead of d pairings as in the original thABE scheme). Moreover, the new scheme enables system engineers to specify various threshold values for distinct sets of attributes. Therefore, this paper describes a practical cryptographic mechanism to support both biometric-based and role-based access control.

Keywords. Fuzzy Identity-Based Encryption, Bio-PKI, Threshold Identity-Based Encryption

1. Introduction

Public key infrastructures (PKIs) are typically used to support confidential and authenticated communications in large electronic systems. When designing a PKI, the following questions must be addressed. How are users distinguished? What public information is required for encryption and signature verification? How are decryption and signing keys generated? Which mechanisms are used to enforce revocation rules? Is private key escrow[1] required ?

In (traditional) certificate-based PKIs (CB-PKIs) [1]: each user (or entity) is assigned a public-key certificate which binds this user's local[2] identifier[3] with a seemingly random number called the user's public key; this public-key certificate and an up-to-date proof of its validity [18] is needed both to encrypt data for the corresponding user and to verify

[1]Correspondence to: Deholo Nali, School of Information Technology and Engineering (SITE), University of Ottawa, 800 King Edward Avenue, P.O. Box 450, Station A Ottawa, Ontario, Canada K1N 6N5. E-mail: deholo@site.uottawa.ca.

[1]Private key escrow refers to the safeguard, by a trusted authority, of a copy of each user's private key.

[2]No practical and general user naming scheme is known, for any class of PKIs.

[3]e.g. the concatenation of the user's legal name and the user's employer name.

signatures associated with this user; each user can autonomously generate her public and private keys; moreover, each user can be required to interact with an online agent (called a SEM) which validates the user's signatures and partially decrypts the user's ciphertexts, if and only if this user is not revoked [3,4]; key escrow is not required.

In identity-based (ID-based) PKIs (IB-PKIs) [5,20]: a trusted authority (TA) generates a set $pubParams$ of system-wide public parameters; only $pubParams$ and a user's local identifier (e.g. email address) are needed both to encrypt messages for this user and to verify signatures associated with this user; the TA generates each user's decryption key, and each user can be required to interact with a SEM [3,4] for both signing and decryption; decryption key escrow is required, but signing key escrow can be avoided [7].

Since the management and distribution of public-key certificates is cumbersome in large and distributed user communities, IB-PKIs are a viable alternative to CB-PKIs, when decryption key escrow is acceptable.

At EuroCrypt'05, Sahai and Waters [19] introduced a novel class of cryptographic schemes called fuzzy *identity*-based encryption (FIBE) schemes. These schemes view identities as sets of descriptive attributes, and use attributes to encrypt messages. More precisely, a user $\mathcal{U}$ with identity ω' is able to decrypt a ciphertext encrypted with identity ω if and only if ω and ω' are within a certain distance from each other, as stipulated by a specifiable metric. One such metric is the *set-overlap* distance, which forces each decryptor of any given ciphertext c to have keys corresponding to a sufficiently large subset of the identity used to obtain c. When a FIBE scheme uses the *set-overlap* distance, this scheme is called a *threshold attributed-based encryption* (thABE) scheme, and thABE schemes are the focus of this paper.

1.1. Applications of thABE to Cryptographic Access Control

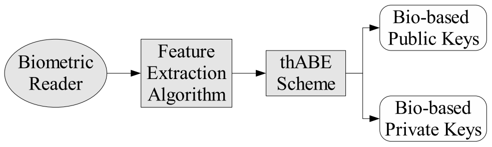


Figure 1. Generation of Cryptographic Keys From Biometrics Using thABE.

thABE schemes are useful for cryptographic access control based on biometrics or roles.

In the case of biometric-based access control, each user is identified by a set of attributes representing features of her biometric identity. These features are extracted by a specialized algorithm (e.g. [8,12]) which takes input from biometric reading devices (e.g. iris scanners). Then, a thABE scheme is used to generate public and private keys from the extracted features (see Figure 1). Each user is given private keys associated with her personal biometric features, and these keys can be used to decrypt ciphertexts encrypted under the user's biometric identity. Thus, access to sensitive information can be controlled via encryption under one's biometric identity (see Figure 2). Since biometric measurements are noisy, the use of thABE significantly improves the possibility of using biometrics to generate cryptographic keys and thereby control access to restricted resources. For instance, a user $\mathcal{A}$ could go to a hospital $\mathcal{H}$ and identify herself via an

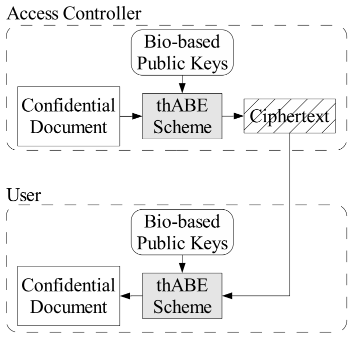


Figure 2. Biometric-Based Access Control Using thABE.

iris scan, *under the ongoing surveillance of a trained agent*. $\mathcal{H}$ could then use this scan to encrypt $\mathcal{A}$'s information when the information needs to be securely sent to $\mathcal{A}$ (e.g. via the Web, or via storage of ciphertexts on a smart card presented by (a representative of) $\mathcal{A}$). In order to obtain her biometric private keys, $\mathcal{A}$ would have to go in person to a trusted third party (e.g. a state agency) which would deliver keys via the same authenticating procedure as that used by $\mathcal{H}$. (Note that such a procedure *significantly reduces the possibility of user impersonation* in the process of obtaining attribute keys.) $\mathcal{A}$ could then decrypt ciphertexts addressed to her, using a thABE scheme.

thABE can also be used for role-based access control (RBAC). Suppose for instance that the CEO of an organization $\mathcal{O}$ wants to send a strategic plan to all the vice presidents of $\mathcal{O}$. The CEO could use a thABE scheme and encrypt the strategic plan under the attributes "*employee-of-$\mathcal{O}$*", "*vice-president*", and "*not-on-vacation*". If the threshold parameter of the thABE scheme is set to 3, then only the vice presidents of $\mathcal{O}$ who are not on vacation will be able to decrypt the ciphered strategic plan. We emphasize that, in practice, RBAC often requires the possession of multiple *specific* roles in order to access confidential documents. In other words, practical RBAC often demands the possession of ℓ out of ℓ (instead of d out ℓ) roles. Therefore, the use of thABE with maximal threshold value is better (i.e. more space and time efficient) than sequentially encrypting documents under all the attributes required to access these documents[4].

1.2. Difficulty of Designing Efficient and Flexible thABE Schemes

Designing efficient and flexible thABE schemes is not trivial. Let d be the threshold parameter of such a scheme (say New-thABE). The following challenges must be simultaneously addressed.

C1 New-thABE should be resistent to collusion attacks whereby users having various attribute keys attempt to pool their keys in order to decrypt ciphertexts which none of them could decrypt on her own. This means that private attribute keys should be "personalized" while public attribute keys should not be tied with users, so that New-thABE be scalable to large user populations.

[4]Indeed, ciphertexts issued by the thABE scheme presented by Sahai and Waters are shorter than ciphertexts obtained by sequentially applying Boneh and Franklin's identity-based (ID-based) encryption (IBE) scheme.

C2 Encryption for an identity ω should target each of the possible sufficiently large subsets of ω. If ω has ℓ elements, then there are $\frac{\ell!}{d!(\ell-d)!}$ possible combinations. However, the cost of encryption should remain low (i.e. ideally constant, but at most linear with respect to d).

C3 A user with identity ω should be given private keys in such a way that at least d of them must be jointly used in order to decrypt each ciphertext. While this could, at first sight, be achieved using a secret sharing scheme, each key must be bound to a specific attribute which makes the problem challenging because the secret sharing scheme should be efficiently combined with a scheme which distinguishes attributes keys.

C4 Encryptors should be able to select attributes from multiple sets whose threshold values are different.

With respect to C3, remark that the decryption procedure of Sahai and Waters'scheme [19] requires d pairing (i.e. expensive) operations. Consequently, one is interested in a scheme whose decryption algorithm involves a small constant number of pairing computations. Remark also that C4 deals with a very practical issue because biometric measurements often include different sets of attributes (instead of only one set of equally important attributes). In order to address C4, Sahai and Waters suggested two methods. First, it was suggested to use different thABE systems with different threshold values. Second, it was proposed to use a maximal threshold value for all ciphertexts, and to use dummy attributes (whose secret keys are given by default to all decryptors) in order to decrease the effective threshold value associated with a given ciphertext. The first proposed option is not necessarily convenient, but most importantly, it increases the length of ciphertexts associated with multiple sets of attributes (by a factor equal to the number of sets). The second proposed option is not necessarily adequate either, as illustrated by the following example. Suppose that $d_{max} = 4$, and that one wants to encrypt a message m for an identity $\omega = \{\alpha_1, \alpha_2\}$ with threshold value $d = 2$. Let δ_1 and δ_2 be two dummy attributes. If m is encrypted for ω with δ_1 and δ_2, then the recipient can decrypt the ciphertext without having the secret keys associated with α_1 and α_2. Moreover, if m is encrypted for with ω with δ_1, then the recipient only needs to have the secret key associated with α_1 or α_2. Thus, while Sahai and Waters addressed C1 and C2 simultaneously, designing a scheme which efficiently addresses C1 through C4 remains an open question. The goal of this paper is to address each of the aforementioned challenges. Our contributions are presented in §1.4, while §1.3 reviews past research related to thABE.

1.3. Related Work

Fuzzy IBE schemes form a class of ID-based encryption schemes. In ID-based cryptography, private keys are secretly sent to users from trusted entities, upon the validation of user identities. As mentioned above, FIBE was recently introduced [19]. As a first FIBE scheme, Sahai and Waters presented a threshold attribute-based encryption scheme. This scheme is distinct from known *threshold IBE* (thIBE) schemes [2,3,14] which view each identity as one string (instead of a set of attribute strings) and which require one decryptor to interact with a threshold number of servers in order to complete decryption procedures. Yao et al. [21] proposed a collusion-resistant ID-based encryption scheme which encrypts to multiple hierarchical identities, but this scheme is computationally expensive (the number of pairings in the decryption procedure grows linearly with the number of hierarchical identities used to encrypt a message.) Much research dealing with the use of biometrics to derive cryptographic secret keys has also been conducted [6,9,13,15,16,17]. In these systems, as pointed out by Sahai and Waters [19], the capture of one's biometric reading enables full impersonation of the corresponding person. This stands in *striking contrast* with Fuzzy IBE which requires each decryptor to obtain private keys, via a biometric measurement performed under the (public) surveillance of a trained agent.

1.4. Contributions

Our contributions in this paper are threefold: the first two contributions deal with practical issues, and the last one is of theoretical importance.

First, we design a thABE scheme which significantly improves the computational requirements of Sahai and Waters'scheme [19] (henceforth referred to as SW-thABE). More precisely, let d be the threshold value of SW-thABE. Then, the decryption procedure of our scheme involves only two pairings, compared with d pairings in the case of SW-thABE. This is achieved at the cost of d extra group additions which are significantly less computationally expensive than pairings. Second, we extend our efficient thABE scheme in order to handle cases in which multiple sets of attributes with different threshold values need to be used. This makes the resulting scheme more applicable to practical situations. Third, we prove that the proposed scheme achieves semantic security with respect to *adaptive*[5] chosen plaintext attacks, in the random oracle model, under the bilinear computational Diffie-Hellman assumption.

1.5. Outline

The sequel is organized as follows: §2 reviews standard terminology concerning thABE and bilinear pairing. §3 describes our proposed thABE scheme, and §4 discusses its storage and computational requirements, in comparison with SW-thABE. §5 summarizes the security guarantees of our scheme, and §6 concludes the paper.

2. Preliminaries

In this section, we present fundamental definitions related to thABE, bilinear pairing and the related standard number theoretic assumptions. Readers familiar with [19] may go to §3.

2.1. Identities

For the description of thABE schemes, each *identity* is viewed as a set ω of *attributes*. Each user with identity ω is given a set of private *attribute keys* each of which corresponds to an element of ω. These attributes keys are secretly granted by an entity known as the private key generator (PKG), upon careful inspection of users'identities (e.g. via surveillance of on-site biometric measurement).

[5]In [19], SW-thABE is proved to offer a weaker security guarantee in the *standard* model (i.e. not the random oracle model (ROM)). The ROM typically allows to obtain better security results than the standard model. However, the former model is less realistic than the latter one.

2.2. Threshold Attribute-Based Encryption Scheme

Each thABE scheme is composed of four algorithms whose functions are described below:

1. `Setup`: Given a security parameter k and a threshold parameter d, this algorithm is used by the PKG to return a tuple *params* of system parameters. These parameters include a description of the message space $\mathcal{M}$ and the ciphertext space $\mathcal{C}$, along with a secret piece of data s_0 called the *master secret key*. Other parameters are allowed, such as *attribute* parameters. Some parameters may be public (including those describing $\mathcal{M}$ and $\mathcal{C}$), while others remain secret (including the master secret key).
2. `Key Generation`: Given the scheme's public and private parameters, and an arbitrary identity ω, this algorithm returns a set D_ω of private keys corresponding to ω.
3. `Encrypt`: Given a message $m \in \mathcal{M}$, the identity ω of an intended recipient, and the scheme's public parameters, this algorithm returns a ciphertext $c \in \mathcal{C}$ corresponding to m and ω.
4. `Decrypt`: Given a ciphertext $c \in \mathcal{C}$, the private key set $D_{\omega'}$ of a recipient, and the scheme's public parameters, this algorithm returns the message $m \in \mathcal{M}$ associated with c.

Encryption and decryption must satisfy the following consistency constraint:

$$\forall\, m \in \mathcal{M}\ \text{Decrypt}(c, pubParams, D_{\omega'}) = m,$$
$$\text{if } |\omega \cap \omega'| \geq d \text{ and } c = \text{Encrypt}(m, \omega, pubParams).$$

2.3. Bilinear Pairing and Diffie-Hellman Problems

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two Abelian groups of prime order q, where $\mathbb{G}_1$ is additive and $\mathbb{G}_2$ is multiplicative. Let $P_0^{(1)} \in \mathbb{G}_1^*$ be a generator of $\mathbb{G}_1$. A *Bilinear pairing* $\hat{e}$ is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ such that $\hat{e}(aP_0^{(1)}, bP_0^{(1)}) = \hat{e}(P_0^{(1)}, P_0^{(1)})^{ab}$ for all $a, b \in \mathbb{Z}_q^*$. (Bilinear pairings can be constructed using *Weil pairings* (cf. section 5 of [5]) and – more efficiently – using *Tate pairings* on elliptic curves.) The map $\hat{e}$ is said to be an *admissible pairing* if it is a *non-degenerate* (i.e. $\hat{e}$ does not send all pairs of points in $\mathbb{G}_1 \times \mathbb{G}_1$ to the identity in $\mathbb{G}_2$), *computable* (i.e. $\hat{e}$ efficiently computes the image of any pair of points in $\mathbb{G}_1 \times \mathbb{G}_1$) *Bilinear pairing*. Let $\mathcal{A}$ be an attacker modelled as a probabilistic Turing machine.

The *computational Diffie-Hellman* (*CDH*) problem [5] is that in which $\mathcal{A}$ is to compute $abP_0^{(1)}$, given $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)})$ and a security parameter k, where $a, b \in \mathbb{Z}_q^*$ are unknown. The success (or *advantage*) of $\mathcal{A}$ is then defined as $Pr[\mathcal{A}$ computes $abP_0^{(1)}]$. The *decisional Diffie-Hellman* (*DDH*) problem [5] is that in which $\mathcal{A}$ is to guess whether $cP_0^{(1)} = abP_0^{(1)}$, given $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)}, cP_0^{(1)})$ and a security parameter k, where $a, b, c \in \mathbb{Z}_q^*$ are unknown. The success (or *advantage*) of $\mathcal{A}$ is then defined as $Pr[\mathcal{A}$ makes a right guess that $cP_0^{(1)} = abP_0^{(1)}]$. $\mathbb{G}_1$ is called a *Gap-Diffie-Hellman group* if the CDH is intractable in $\mathbb{G}_1$, but the the DDH can be solved in polynomial time in $\mathbb{G}_1$. The *Bilinear Diffie-Hellman* (*BDH*) problem [5] is that in which $\mathcal{A}$ is to compute $\hat{e}(P_0^{(1)}, P_0^{(1)})^{abc}$ given a security parameter k and the tuple $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)}, cP_0^{(1)})$, where $a, b, c \in \mathbb{Z}_q^*$ are unknown. The success (or *advantage*) of $\mathcal{A}$ is then defined as $Pr[\mathcal{A}$ outputs $\hat{e}(P_0^{(1)}, P_0^{(1)})^{abc}]$.

3. Proposed Encryption Scheme

For increased clarity, we present our thABE scheme in two steps. First, we describe the core of our new scheme (New-thABE-v1) which simultaneously solves challenges C1, C2 and C3. Second, we extend New-thABE-v1 so that it handles multiple attribute sets whose threshold values are different from one another. This extended scheme (which addresses challenge C4) is named New-thABE-v2.

3.1. Fundamental Efficient Scheme – New-thABE-v1

1. `Instance Generator` (k). This procedure, denoted by IG, is a randomized algorithm which takes a security parameter $k > 0$, runs in $O(k)$, and outputs $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$, where $\mathbb{G}_1$ and $\mathbb{G}_2$ are two Abelian groups of prime order $q \geq 2^k$, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is an admissible pairing with respect to which $\mathbb{G}_1$ and $\mathbb{G}_2$ are Gap-Diffie-Hellman groups.
2. `Setup` (k, d): Given a security parameter $k > 0$, the *PKG*:

   (a) runs IG with input k and obtains $(\mathbb{G}_1, \mathbb{G}_2, \hat{e})$.
   (b) computes $n = poly_1(k)$ and $\ell = poly_2(k)$, where $poly_1$ and $poly_2$ are polynomials over the positive integers (n is the message length and ℓ is the total number of attributes);
   (c) picks, randomly and uniformly[6], $P_0^{(1)}, P_0^{(2)} \in \mathbb{G}_1$ and $s_0, a_1, a_2, \cdots, a_\ell \in \mathbb{Z}_q^*$;
   (d) computes $g = \hat{e}(P_0^{(2)}, s_0 P_0^{(1)})$ and $A_i = a_i P_0^{(1)}$ for $1 \leq i \leq \ell$;
   (e) chooses a cryptographic hash function $\mathcal{H}_2 : \mathbb{G}_2 \rightarrow \{0,1\}^n$.

   The message space is $\mathcal{M} = \{0,1\}^n$ and the ciphertext space is $\mathcal{C} = \{0,1\}^\ell \times \mathbb{G}_1^{\ell+2} \times \{0,1\}^n$. The system's public parameters (which must be certified) are $pubParams = (q, n, \hat{e}, P_0^{(1)}, P_0^{(2)}, g, \mathcal{H}_2, (A_i)_{i=1}^{\ell})$, and the *PKG* keeps s_0 and $(a_i)_{i=1}^{\ell}$ secret, while $params = (pubParams, s_0, (a_i)_{i=1}^{\ell})$.
3. `Key Generation` $(\omega, params)$: Let $\omega = \{i_j\}_{j=1}^{t}$ be the identity of a user $\mathcal{U}$, where each i_j is the index of an attribute. Then, the PKG:

   (a) selects a random polynomial $q_{\mathcal{U}}(x) = s_0 + \sum_{z=1}^{d-1} r_{(u,z)} x^z$ (where $r_{(u,z)} \in_R \mathbb{Z}_q^*$ for $1 \leq z \leq d-1$);
   (b) computes and secretly gives $\mathcal{U}$ her private attribute key set $D_\omega = \{D_{i_j}\}_{j=1}^{t}$, where $D_{i_j} = (q_{\mathcal{U}}(i_j) + a_i) P_0^{(2)}$ for $1 \leq j \leq t$.
4. `Encrypt` $(m, \omega, pubParams)$: Given a message $m \in \mathcal{M}$, the identity $\omega = \{i_j\}_{j=1}^{t}$ of an intended recipient, and $pubParams$, an the encryptor:

[6] In the sequel, we shall use the notation $x \in_R X$ to indicate that the element x is chosen uniformly at random from the set X.

(a) picks $r_1, r_2 \in_R \mathbb{Z}_q^*$;
(b) computes $U_1 = r_1 P_0^{(1)}$, $U_2 = r_2 P_0^{(2)}$, and $(V_{i_j})_{j=1}^t$, where $V_{i_j} = \frac{r_1}{r_2} A_{i_j}$ for $1 \leq j \leq t$;
(c) computes the bit string $\sigma \in \{0,1\}^\ell$ whose e^{th} bit is 1 if and only if $e = i_j$ for some i_j, where $\omega = \{i_j\}_{j=1}^t$;
(d) computes $W = m \oplus \mathcal{H}_2(g^{r_1})$ and outputs $c = (\sigma, U_1, U_2, V_{i_1}, V_{i_2}, \cdots, V_{i_t}, W)$.

5. `Decrypt` $(c, D_{\omega'}, pubParams)$: Let $c = (\sigma, U_1, U_2, V_{i_1}, V_{i_2}, \cdots, V_{i_t}, W)$ be a given ciphertext intended for the identity ω, $D_{\omega'}$ be a recipient's private key set, and Φ be any d-element subset of $\omega \cap \omega'$. Then, the recipient outputs

$$m = W \oplus \mathcal{H}_2 \left(\hat{e}(\sum_{i \in \Phi} \phi_i D_i, U_1) \cdot \hat{e}(U_2, \sum_{i \in \Phi} \phi_i V_i)^{-1} \right),$$

where each $\phi_i = \prod_{j \in \Phi \setminus \{i\}} \frac{-j}{i-j}$ is the zero evaluation of the Lagrange coefficient of i with respect to Φ.

For correctness, remark that:

$$\begin{aligned}
\hat{e}(\sum_{i \in \Phi} \phi_i D_i, U_1) &= \hat{e}((\sum_{i \in \Phi} \phi_i q_{\mathcal{U}}(i)) P_0^{(2)} + (\sum_{i \in \Phi} \phi_i a_i) P_0^{(2)}, r_1 P_0^{(1)}) \\
&= \hat{e}(s_0 P_0^{(2)}, r_1 P_0^{(1)}) \cdot \hat{e}(P_0^{(2)}, (\sum_{i \in \Phi} \phi_i a_i) r_1 P_0^{(1)}) \\
&= \hat{e}(P_0^{(2)}, s_0 P_0^{(1)})^{r_1} \cdot \hat{e}(r_2 P_0^{(2)}, \sum_{i \in \Phi} \phi_i (\frac{r_1}{r_2} A_i)) \\
&= g^{r_1} \cdot \hat{e}(U_2, \sum_{i \in \Phi} \phi_i V_i)
\end{aligned}$$

3.2. Dealing with Multiple Attribute Sets – New-thABE-v2

The scheme which is presented below assumes the existence of two attribute sets Λ_1 and Λ_2 whose threshold values are d_1 and d_2 respectively. A more general scheme with polynomially many attribute sets could also be presented, using the very same technique. However, for simplicity, we only describe the case in which there are two sets.

1. `Instance Generator` (k). Same as in New-thABE-v1.
2. `Setup` (k, d_1, d_2): Same as in New-thABE-v1, except for the following changes:

(a) Instead of ℓ, the PKG computes $\ell_1 = poly_2(k)$ and $\ell_2 = poly_3(k)$, $poly_2$ and $poly_3$ are polynomials over the positive integers (ℓ_i is the total number of attributes in Λ_i for $i = 1, 2$).
(b) In addition to $(a_i)_{i=1}^{\ell_1}$, the PKG also defines $(b_i)_{i=1}^{\ell_2}$ (where each $b_i \in_R \mathbb{Z}_q^*$).
(c) In addition to $(A_i)_{i=1}^{\ell_1}$, the PKG also defines $(B_i)_{i=1}^{\ell_2}$ (where each $B_i = b_i P_0^{(1)}$).

The message space $\mathcal{M}$ remains the same as in New-thABE-v1, but the ciphertext space is now set to $\mathcal{C} = \{0,1\}^{\ell_1+\ell_2} \times \mathbb{G}_1^{\ell_1+\ell_2+2} \times \{0,1\}^n$. $pubParams = (q, n, \hat{e}, P_0^{(1)}, P_0^{(2)}, g, \mathcal{H}_2, (A_i)_{i=1}^{\ell}, (B_i)_{i=1}^{\ell})$, and $params = (pubParams, s_0, (a_i)_{i=1}^{\ell_1}, (b_i)_{i=1}^{\ell})$.

3. `Key Generation` $(\omega, params)$: Let $\omega = \{i_j\}_{j=1}^{t_1} \cup \{u_v\}_{v=1}^{t_2}$ be the identity of a user $\mathcal{U}$, where each i_j is the index of an attribute in Λ_1 and each u_v is the index of an attribute in Λ_2. Then, the PKG:

   (a) selects two random polynomial $q_{(\mathcal{U},1)}(x)$ and $q_{(\mathcal{U},1)}(x)$, where $q_{(\mathcal{U},i)}(x) = s_0 + \sum_{z_i=1}^{d_i-1} r_{(u,z_i)} x^{z_i}$ (with $r_{(u,z_i)} \in_R \mathbb{Z}_q^*$ for $1 \le z_i \le d_i - 1$) for $i = 1, 2$.
   (b) computes and secretly gives $\mathcal{U}$ her private attribute key set $D_\omega = \{D_{(i_j,1)}\}_{j=1}^{t_1} \cup \{D_{(u_v,2)}\}_{v=1}^{t_2}$, where $D_{(i_j,1)} = (q_{(\mathcal{U},1)}(i_j) + a_i)P_0^{(2)}$ for $1 \le j \le t_1$ and $D_{(u_v,2)} = (q_{(\mathcal{U},2)}(u_v) + b_i)P_0^{(2)}$ for $1 \le j \le t_2$.

4. `Encrypt` $(m, \omega, pubParams)$: Given a message $m \in \mathcal{M}$, $pubParams$, and the identity $\omega = \{i_j\}_{j=1}^{t_1} \cup \{u_v\}_{v=1}^{t_2}$ of an intended recipient (where each i_j is the index of an attribute in Λ_1 and each u_v is the index of an attribute in Λ_2), the encryptor:

   (a) picks $r_1, r_2 \in_R \mathbb{Z}_q^*$ and computes U_1, U_2, $(V_{i_j})_{j=1}^{t}$, and W as in New-thABE-v1;
   (b) computes $(X_{u_v})_{v=1}^{t_2}$ where $X_{u_v} = \frac{r_1}{r_2} B_{u_v}$ for $1 \le v \le t_2$;
   (c) computes $\sigma_1 \in \{0,1\}^{\ell_1}$ and $\sigma_2 \in \{0,1\}^{\ell_2}$ such that the e^{th} bit of σ_1 is 1 if and only if $e = i_j$ (for some i_j in $\{i_j\}_{j=1}^{t_1}$), and the e^{th} bit of σ_2 is 1 if and only if $e = u_v$ (for some u_v in $\{u_v\}_{v=1}^{t_2}$);
   (d) outputs $c = (\sigma_1, \sigma_2, U_1, U_2, (V_{i_j})_{j=1}^{t_1}, (X_{u_v})_{v=1}^{t_2}, W)$.

5. `Decrypt` $(c, d_{\omega'}, pubParams)$: Let $c = (\sigma_1, \sigma_2, U_1, U_2, (V_{i_j})_{j=1}^{t_1}, (X_{u_v})_{v=1}^{t_2}, W)$ be a given ciphertext intended for the identity ω and $D_{\omega'}$ be a recipient's private key set. For $i = 1, 2$, let Φ_i be a d_i-element subset of $\omega \cap \omega'$ such that Φ_i only includes indexes of Λ_i attributes. Then, the recipient outputs $m = W \oplus \mathcal{H}_2(V)$, where

$$V = (\hat{e}(\sum_{i \in \Phi_1} \phi_i D_{(i,1)} + \sum_{i \in \Phi_2} \phi_i D_{(i,2)}, U_1) \cdot \hat{e}(U_2, \sum_{i \in \Phi_1} \phi_i V_i + \sum_{i \in \Phi_2} \phi_i X_i)^{-1})^{\frac{1}{2}}.$$

4. Efficiency

New-thABE and SW-thABE have the same storage requirements: if a user 's identity has δ attributes, then this user is given δ attribute decryption keys.

Table 1 compares the computational requirements of New-thABE with those of SW-thABE. d denotes the (sum of all) threshold value(s), while t denotes both number of attributes used for encryption and the number of attributes which identify an arbitrary user. M_X and A_X respectively denote computational costs of scalar multiplication and addition in the Abelian group X. R_X denotes the computational cost of uniformly selecting a random element in the set X. The computational cost of exponentiation in the group X is denoted by Ex_X, and P denotes the computational cost of a bilinear pairing oper-

Table 1. Efficiency Comparison of New-thABE with SW-thABE.

Schemes		**Computational Requirements**	**Features**
SW-thABE	Public Parameters	$Ex_{\mathbb{G}_2} + \ell \cdot M_{\mathbb{G}_1} + P + R_{\mathbb{G}_1} + (\ell+1)R_{\mathbb{Z}_q^*}$	1 Attribute Set
	Key Generation	$(t \cdot d)A_{\mathbb{Z}_q^*} + (t \cdot d)Ex_{\mathbb{Z}_q^*} + t \cdot Inv_{\mathbb{Z}_q^*} + t \cdot M_{\mathbb{G}_1} + (t \cdot d)M_{\mathbb{Z}_q^*} + d \cdot R_{Zqs}$	1 Threshold Value
	Encryption	$Ex_{\mathbb{G}_2} + t \cdot M_{\mathbb{G}_1} + R_{\mathbb{Z}_q^*}$	
	Decryption	$(d+1)Ex_{\mathbb{G}_2} + Inv_{\mathbb{G}_2} + M_{\mathbb{G}_2} + d \cdot P$	
New-thABE	Public Parameters	$Ex_{\mathbb{G}_2} + \ell \cdot M_{\mathbb{G}_1} + P + 2R_{\mathbb{G}_1} + (\ell+1)R_{\mathbb{Z}_q^*}$	Many Attribute Sets
	Key Generation	$t(d+1)A_{\mathbb{Z}_q^*} + (t \cdot d)Ex_{\mathbb{Z}_q^*} + t \cdot M_{\mathbb{G}_1} + t(d-1)M_{\mathbb{Z}_q^*} + d \cdot R_{Zqs}$	Flexible
	Encryption	$Ex_{\mathbb{G}_2} + t \cdot Inv_{\mathbb{Z}_q^*} + t \cdot M_{\mathbb{Z}_q^*} + (t+2)M_{\mathbb{G}_1} + 2R_{\mathbb{Z}_q^*}$	Threshold Values
	Decryption	$2(d-1)A_{\mathbb{G}_1} + Ex_{\mathbb{G}_2} + Inv_{\mathbb{G}_2} + 2d \cdot M_{\mathbb{G}_1} + M_{\mathbb{G}_2} + 2P$	

ation. Moreover, neither the computational cost of hash functions, nor that of exclusive OR operations are taken into account.

Table 1 shows that the *Setup* and the *Encryption* procedures of SW-thABE and New-thABE have similar computational requirements. Note however that New-thABE's *Encryption* algorithm requires t more inversions and multiplications in $\mathbb{Z}_q^*$ than SW-thABE's *Encryption* procedure. This difference is nonetheless negligible because $tM_{\mathbb{G}_1}$ is significantly greater than $t(M_{Zqs} + M_{\mathbb{Z}_q^*})$. Moreover, New-thABE's *Key Generation* procedure is slightly more efficient than SW-thABE's, due to the absence of inversions in the former. The main advantage of New-thABE stems from its decryption procedure which essentially replaces pairings with additions in $\mathbb{G}_1$ (these additions are significantly less expensive than pairings). Furthermore, Table 1 emphasizes that New-thABE handles multiple attributes sets whose threshold values can be specified.

5. Security

This section presents the security guarantees of our proposed thABE scheme. The notions of adaptive chosen ciphertext security and chosen plaintext security for threshold ABE systems are defined in §5.1 and §5.2. Then, §5.3 shows that New-thABE-v1 is semantically secure with respect to adaptive chosen plaintext attacks, in the random oracle model, if the BDH problem is hard.

5.1. Chosen Ciphertext Security

Let Ψ be a thABE scheme. Then, the following game, initiated by a challenger Ch against an attacker $\mathcal{A}$, may then be considered:

Setup: From a security parameter k and a threshold parameter d, $\mathcal{C}h$ uses Ψ's *Setup* algorithm to generate the cryptosystem's public and private parameters - keeping the private parameters secret while giving the public ones to $\mathcal{A}$.
Phase 1: $\mathcal{A}$ issues to $\mathcal{C}h$ a polynomially bounded number of queries of the following types:

$*$ *Key Extraction query*: Given an identity ω_i, $\mathcal{C}h$ must return the private key set D_{ω_i} associated with ω_i.
$*$ *Decryption query*: Given a private key set $D_{\omega'_i}$, and a ciphertext c_i encrypted for an identity ω_i such that $|\omega_i \cap \omega'_i| \geq d$, $\mathcal{C}h$ must return a message m_i associated with c_i.

The above queries may be issued adaptively: each request may depend on its predecessors.
Challenge: Once *Phase 1* is over, $\mathcal{A}$ issues an identity ω^* of its choice, and a pair (m_0, m_1) of equal-length plaintexts, such that, in *Phase 1*, no *Key Extraction* queries were issued on an identity ω such that $|\omega \cap \omega^*| \geq d$. $\mathcal{C}h$ then picks a random bit $\beta \in \{0, 1\}$, computes the encryption c^* of m_β for ω^*, and sends c^* to $\mathcal{A}$.
Phase 2: $\mathcal{A}$ issues a polynomially bounded number of *Phase 1* types of queries, under the following restrictions:

$*$ No *Key Extraction* queries are issued on an identity ω_i such that $|\omega_i \cap \omega^*| \geq d$.
$*$ If a *Decryption* query is issued with c^* as an argument, then the corresponding identity ω_i cannot satisfy $|\omega_i \cap \omega^*| \geq d$.

The queries may be issued adaptively: each request may depend on its predecessors.
Guess: Once *Phase 2* is over, $\mathcal{A}$ submits a guess bit β' and wins the game if $\beta' = \beta$.

Definition 1 *The above game is called the* IND-thABE-CCA *attack game. The quantity* $|Pr[\beta' = \beta] - \frac{1}{2}|$ *– representing the advantage of* $\mathcal{A}$ *over any challenger* $\mathcal{C}h$ *in the game – is denoted by* $Adv^{thABE-CCA}_{\mathcal{A},\mathcal{C}h}$. *A thABE scheme is said to be secure against adaptive chosen ciphertext attacks (*IND-thABE-CCA *secure, in short) if no polynomially bounded attacker can be found that has non-negligible advantage in the above* IND-thABE-CCA *game.*

5.2. Chosen Plaintext Security

The notion of chosen plaintext security is similar to (and weaker than) the notion of *chosen ciphertext security*. To define semantic security for thABE schemes, one may consider a game (called the *IND-thABE-CPA* game), which is identical to the *IND-thABE-CCA* game, except that the attacker $\mathcal{A}$ is unable to issue *Decryption* queries.

Definition 2 *The value* $|Pr[\beta' = \beta] - \frac{1}{2}|$ *– representing the advantage of* $\mathcal{A}$ *over any challenger* $\mathcal{C}h$ *in the* IND-thABE-CPA *game – is denoted by* $Adv^{thABE-CPA}_{\mathcal{A},\mathcal{C}h}$. *A thABE scheme is said to be secure against adaptive chosen plaintext attacks (*IND-thABE-

CPA *secure, in short) if no polynomially bounded attacker can be found that has non-negligible advantage in a* IND-thABE-CPA *game.*

5.3. Security Theorem

Theorem 1 *Let k be a security parameter k. Assume that the hash function of New-thABE-v1 is a random oracle. Suppose also that there exists an attacker $\mathcal{A}$ which has non-negligible advantage $\varepsilon(k)$, in time τ, against any challenger $\mathcal{C}h$, in the* IND-thABE-CPA *game. Then, there exists an algorithm $\mathcal{B}$ which solves the BDH problem, in time $O(\tau)$, with non-negligible advantage at least $\frac{\varepsilon(k)}{q_{\mathcal{H}_2}}$, where $q_{\mathcal{H}_2}$ is the number of $\mathcal{H}_2$ queries issued by $\mathcal{A}$ in the attack game.*

See the Appendix for the **proof** of Theorem 1. Moreover, note that, in [5], it is shown how to prove that the FullIndent scheme is IND-ID-CCA secure using the fact that the BasicIdent scheme is IND-ID-CPA secure, where FullIdent is the obtained from BasicIdent by applying the so-called Fujisaki-Okamoto padding [10]. In the same way, New-thABE can be transformed into a scheme which is IND-thABE-CCA secure. This can be done as follows: (1) transform New-thABE-v1 into a scheme Basic-New-thABE-Pub (using the method described in [5] to transform BasicIdent into BasicPub); (2) show that Basic-New-thABE-Pub is IND-CPA if the BDH is intractable (using the method described in the proof of Theorem 1); (3) show that New-thABE-v1 is IND-ID-CPA if Basic-New-thABE-Pub is IND-CPA (using the method described in the proof of Lemma 4.2 of [5]); (4) transform Basic-New-thABE-Pub into a scheme Full-New-thABE-Pub (by applying Fujisaki-Okamoto padding [10]); transform Full-New-thABE-Pub into a scheme Full-New-thABE (as FullIdentPub is transform into FullIdent in [5]); show that Full-New-thABE is IND-ID-CCA if Full-New-thABE-Pub is IND-CCA (using the method described in the proof of Lemma 4.6 of [5]); apply Lemma 4.5 of [5] to prove that Full-New-thABE-Pub is IND-CCA if Basic-New-thABE-Pub is IND-CPA; conclude that Full-New-thABE is IND-ID-CCA secure if the BDH problem is intractable. (Note that the above line of reasoning has been considered standard in the literature [11,19].)

6. Conclusion

The aim of this paper was to describe an efficient threshold attribute-based encryption (thABE) scheme which handles multiple attribute sets with specifiable threshold values. Applications of such a scheme include *biometric-based* and *role-based* cryptographic access control. Building bio-PKIs is a challenging task, and we believe that our scheme provides a practical solution to this challenge. To the best of our knowledge, the proposed scheme is the most efficient one of its class. In particular, the new scheme is significantly more efficient than Sahai and Waters'scheme. Moreover, the security of our scheme was studied, showing that it achieves semantic security with respect to chosen plaintext attacks (in the random oracle model, under a standard number theoretic assumption).

For future work, we note that building a threshold ABE scheme enabling system users to specify threshold values *at encryption time* remains an open question. (Our scheme only allows system engineers to specify *system-wide* threshold values for various sets).

Acknowledgements

This work was partially supported by grants from the Natural Sciences and Engineering Research Council of Canada (NSERC). The authors would like to thank anonymous reviewers of IWAP'05, for insightful comments on a previous version of this paper.

References

[1] C. Adams and S. Lloyd, *Understanding PKI: Concepts, Standards, and Deployment Considerations, Second Edition*, Addison-Wesley, 2002.

[2] J. Baek and Y. Zheng, *Identity-Based Threshold Decryption*, Proceedings of the 7th International Workshop on Theory and Practice in Public Key Cryptography (PKC'04), Lecture Notes in Computer Science, vol. 2947, Springer-Verlag, 2004, pp. 262–276.

[3] D. Boneh, X. Ding, and G. Tsudik, *Identity-based Mediated RSA*, Proceedings of the third International Workshop on Information and Security Applications (WISA'02) (Jeju Island, Korea), 2002.

[4] D. Boneh, X. Ding, G. Tsudik, and C. Wong, *A Method for Fast revocation of Public Key Certificates and Security Capabilities*, Proceedings of the 10th USENIX Security Symposium, USENIX, 2001, pp. 297–308.

[5] D. Boneh and M.K. Franklin, *Identity-Based Encryption from the Weil Pairing*, Proceedings of the 21st Annual International Cryptology Conference on Advances in Cryptology, vol. 2139, Springer-Verlag, 2001, pp. 213–229.

[6] X. Boyen, *Reusable Cryptographic Fuzzy Extractors*, ACM Conference on Computer and Communications Security (CCS'04), ACM Press, 2004, Available at `http://www.cs.stanford.edu/~xb/ccs04/`, pp. 82–91.

[7] X. Chen, F. Zhang, and K. Kim, *A new id-based group signature scheme from bilinear pairings*, 2003, `http://eprint.iacr.org/`.

[8] J. G. Daugman, *How Iris Recognition Works*, IEEE Transaction on Circuits and Systems for Video Technology **14** (2004), no. 1, 21–30.

[9] Y. Dodis, L. Reyzin, and S. Smith, *Fuzzy Extractor: How to generate string keys from biometrics and other noisy data*, Proceedings of EUROCRYPT'04 on Advances in Cryptology, Lecture Notes in Computer Science, vol. 3027, Springer-Verlag, 2004, pp. 523–540.

[10] E. Fujisaki and T. Okamoto, *Secure Integration of Asymmetric and Symmetric Encryption Schemes*, Lecture Notes in Computer Science, vol. 1666, Springer-Verlag, 1999, pp. 537–554.

[11] C. Gentry and A. Silverberg, *Hierarchical ID-Based Cryptography*, Proceedings of the 8th International Conference on the Theory and Application of Cryptology and Information Security, vol. 2501, Springer-Verlag, 2002, pp. 548–566.

[12] A. Jain, A. Ross, and S. Prabhakar, *Fingerprint Matching Using Minutiae and Texture Features*, Proceedings of the International Conference on Image Processing (ICIP), 2001, pp. 282–285.

[13] A. Juels and M. Wattenberg, *A Fuzzy commitment scheme*.

[14] B. Libert and J.-J. Quisquater, *Efficient Revocation and Threshold Pairing Based Cryptosystems*, Proceedings of the twenty-second annual symposium on Principles of distributed computing, ACM Press, 2003, pp. 163–171.

[15] F. Monrose, M. K. Reiter, Q. Li, D. Lopresti, and C. Shih, *Towards voice generated cryptographic keys on ressource constrained devices*, Proceedings of the 11th USENIX Security Symposium, 2002.

[16] F. Monrose, M. K. Reiter, Q. Li, and S. Wetzel, *Cryptographic key generation from voice*, Proceedings of the IEEE Conference on Security and Privacy, IEEE Press, 2001, pp. 202–213.

[17] F. Monrose, M. K. Reiter, and S. Wetzel, *Password hardening based on key-stroke dynamics*, Proceedings of the 6th ACM Conference on Computer and communications security (CCS'99), ACM Press, 1999, pp. 73–82.
[18] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, *Internet public key infrastructure online certificate status protocol - OCSP*, RFC 2560.
[19] A. Sahai and B. Waters, *Fuzzy Identity Based Encryption*, Proceedings of EUROCRYPT'05 on Advances in cryptology, Lecture Notes in Computer Science, vol. 3494, Springer-Verlag, 2005, pp. 457–473.
[20] A. Shamir, *Identity-Based Cryptosystems and Signature Schemes*, Proceedings of CRYPTO'84 on Advances in cryptology, Springer-Verlag New York, Inc., 1984, pp. 47–53.
[21] D. Yao, N. Fazio, Y. Dodis, and A. Lysyanskaya, *ID-Based Encryption for Complex Hierarchies with Applications to Forward Security and Broadcast Encryption*, Proceedings of the ACM Conference on Computer and Communications Security (CCS'04).

Appendix (Proof of Theorem 1)

In this proof, we show how to construct $\mathcal{B}$ using $\mathcal{A}$. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two Abelian groups of prime order q, where $\mathbb{G}_1$ is additive and $\mathbb{G}_2$ is multiplicative. Let $P_0^{(1)} \in \mathbb{G}_1^*$ be a generator of $\mathbb{G}_1$, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear pairing on $\mathbb{G}_1$, such that the BDH is assumed to be hard with respect to $\hat{e}$, and let $(\mathbb{G}_1, q, P_0^{(1)}, aP_0^{(1)}, bP_0^{(1)}, cP_0^{(1)})$ be a tuple for which a, b, c are unknown to $\mathcal{B}$. $\mathcal{B}$'s goal is to solve the BDH Problem by computing $\hat{e}(P_0^{(1)}, P_0^{(1)})^{abc}$ in polynomial time. To achieve this goal, $\mathcal{B}$ initiates an *IND-thABE-CPA* game with $\mathcal{A}$, using an arbitrary security parameter k and threshold parameter d.

$\mathcal{B}$ first sets $n = poly_1(k)$ and $\ell = poly_2(k)$, using the polynomials $poly_1$ and $poly_2$. Then $\mathcal{B}$ picks $y \in_R \mathbb{Z}_q^*$, and sets $P_0^{(2)} = aP_0^{(1)}$, $Y = yP_0^{(2)}$, and $g = g_1 \cdot g_2$ (where $g_1 = \hat{e}(Y, P_0^{(1)})$ and $g_2 = \hat{e}(P_0^{(2)}, bP_0^{(1)})^{-1}$). $\mathcal{B}$ defines a polynomial $f(x) = 1 + \sum_{j=1}^{d-1} \rho_j x^j$ (where $\rho_j \in_R \mathbb{Z}_q^*$ for $1 \leq j \leq d-1$), and sets $c_i = f(i)$ for $1 \leq i \leq \ell$. Remark that $1 = \sum_{i\in\Delta} \phi_i c_i$ for any d-element subset Δ of $\{1, \cdots, \ell\}$ (where $\phi_i = \prod_{j\in\Phi\setminus\{i\}} \frac{-j}{i-j}$). $\mathcal{B}$ picks $a_i \in_R \mathbb{Z}_q^*$ for $1 \leq i \leq \ell$, and sets $A_i = c_i(bP_0^{(1)}) + a_i P_0^{(1)}$ for $1 \leq i \leq \ell$. Note that $\sum_{i\in\Delta} \phi_i A_i = bP_0^{(1)} + (\sum_{i\in\Phi} \phi_i a_i)P_0^{(1)}$ if Δ is defined as above. $\mathcal{B}$ defines a hash function $\mathcal{H}_2^{sim} : \mathbb{G}_2 \to \{0,1\}^n$ over which it has full control, and sets $\mathcal{M} = \{0,1\}^n$, $\mathcal{C} = \{0,1\}^\ell \times \mathbb{G}_1^{\ell+2} \times \{0,1\}^n$, $pubParams = (q, n, \hat{e}, P_0^{(1)}, P_0^{(2)}, g, \mathcal{H}_2, (A_i)_{i=1}^{\ell})$, $params = (pubParams, b, (a_i)_{i=1}^{\ell})$, where b is not known by $\mathcal{B}$.

The rest of the proof consists of three sections. The *Hash Simulation* section shows how $\mathcal{B}$ simulates $\mathcal{H}_2^{sim}$. The *Attack Game* section explains how $\mathcal{B}$ handles the queries of the attack game. Finally, the *Complexity and Probability* section derives the complexity and probability results of the theorem.

Hash Simulation:

For $\mathcal{H}_2^{sim}$-queries, $\mathcal{B}$ maintains a list $L_{\mathcal{H}_2^{sim}}$ whose entries have the form $(\alpha, h_{2,\alpha})$, where $\alpha \in \mathbb{G}_2$ and $h_{2,\alpha} \in \{0,1\}^n$ is the simulated value of $\mathcal{H}_2(\alpha)$. Thus, on input $\alpha \in \mathbb{G}_2$, $\mathcal{B}$ proceeds as follows:

- If $L_{\mathcal{H}_2^{sim}}$ already has an entry $(\alpha, h_{2,\alpha})$, then $\mathcal{B}$ returns $h_{2,\alpha}$ as an answer to the $\mathcal{H}_2^{sim}$-query;

- Otherwise, $\mathcal{B}$ picks $h_{2,\alpha}$ uniformly at random in $\{0,1\}^n$, adds $(\alpha, h_{2,\alpha})$ to $L_{\mathcal{H}_2^{sim}}$, and returns $h_{2,\alpha}$ as an answer to the $\mathcal{H}_2^{sim}$ query.

Attack Game:
$\mathcal{B}$ handles $\mathcal{A}$'s queries as follows:

- *Phase 1*: Given an identity $\omega_i = \{i_j\}_{j=1}^{t_i}$, $\mathcal{B}$ picks $r_{(i,j)} \in_R \mathbb{Z}_q^*$ (for $1 \leq j \leq d-1$), defines a polynomial $q_{\omega_i}(x) = y + \sum_{j=1}^{d-1} r_{(i,j)} x^j$, sets $D_{i_j} = (q_{\omega_i}(i_j) + a_{i_j}) P_0^{(2)}$ (for $1 \leq j \leq t_i$), and returns $D_{\omega_i} = \{D_{i_j}\}_{j=1}^{t_i}$. Note that if $r_1, r_2 \in_R \mathbb{Z}_q^*$, $U_1 = r_1 P_0^{(1)}$, $U_2 = r_1 P_0^{(2)}$, $W = m \oplus \mathcal{H}_2(g^{r_1})$, and $V_j = rA_j$ for all $j \in \Delta$ (where Δ is any d-element subset of ω_i), then $\hat{e}(\sum_{i\in\Delta} \phi_i D_i, U_1) \cdot \hat{e}(U_2, \sum_{i\in\Delta} \phi_i V_i)^{-1} = \hat{e}(yP_0^{(2)}, P_0^{(1)})^{r_1} \cdot \hat{e}(P_0^{(2)}, bP_0^{(1)})^{-1} = g_1^r \cdot g_2^r = g^r$.
- *Challenge Phase*: $\mathcal{A}$ issues an identity $\omega^* = \{i_j^*\}_{j=1}^{t^*}$ of its choice, and a pair (m_0, m_1) of equal-length plaintexts, such that, in *Phase 1*, no *Key Extraction* queries were issued on an identity ω such that $|\omega \cap \omega^*| \geq d$. $\mathcal{C}h$ then picks a random bit $\beta \in \{0,1\}$, $W^* \in_R \{0,1\}^n$, $r_2^* \in_R \mathbb{Z}_q^*$, and sets $U_2^* = r_2^* P_0^{(2)}$, $V_{i_j^*}^* = \frac{1}{r_2^*}(c_{i_j^*}(bP_0^{(1)}) + a_{i_j^*}(cP_0^{(1)}))$ for $1 \leq j \leq t^*$, and $U_1^* = cP_0^{(1)}$. Then $\mathcal{B}$ computes σ^* according to the method described in New-thABE's *Encryption* procedure, and returns $c^* = (\sigma^*, U_1^*, U_2^*, (V_{i_j^*}^*)_{j=1}^{t^*}, W^*)$ to $\mathcal{A}$.
- *Phase 2*: $\mathcal{B}$ proceeds as in *Phase 1*.
- *Guess*: $\mathcal{A}$ sends $\mathcal{B}$ his guess β' and wins if $\beta' = \beta$. Regardless of $\mathcal{A}$'s success, $\mathcal{B}$ picks (uniformly at random) an entry $(\alpha, h_{2,\alpha})$ of $L_{\mathcal{H}_2}$, and submits $(\alpha \cdot \hat{e}(aP_0^{(1)}, bP_0^{(1)}))^{\frac{1}{2}}$ as a solution of the BDH problem.

Complexity and Probability:
Note that $\mathcal{A}$ wins the *IND-thABE-CPA* with non-negligible probability if and only if $\mathcal{A}$ computes $\mathcal{H}_2(\eta)$, where (for some for some d-element subset Φ of ω^*):

$$\begin{aligned}
\eta &= \hat{e}(\sum_{i\in\Phi} \phi_i D_i^*, U_1^*) \cdot \hat{e}(U_2^*, \sum_{i\in\Phi} \phi_i V_i^*)^{-1} \\
&= \hat{e}(bP_0^{(2)}, cP_0^{(1)}) \cdot \hat{e}(\sum_{i\in\Phi} \phi_i(c_i b + a_i)P_0^{(2)}, cP_0^{(1)}) \cdot \hat{e}(U_2^*, \sum_{i\in\Phi} \phi_i V_i^*)^{-1} \\
&= \hat{e}(P_0^{(1)}, P_0^{(1)})^{abc} \cdot \hat{e}(bP_0^{(2)}, cP_0^{(1)}) \cdot \hat{e}(\sum_{i\in\Phi} \phi_i a_i P_0^{(2)}, cP_0^{(1)}) \cdot \hat{e}(U_2^*, \sum_{i\in\Phi} \phi_i V_i^*)^{-1} \\
&= \hat{e}(P_0^{(1)}, P_0^{(1)})^{2(abc)} \cdot \hat{e}(\sum_{i\in\Phi} \phi_i a_i P_0^{(2)}, cP_0^{(1)}) \cdot \hat{e}(r_2^* P_0^{(2)}, \sum_{i\in\Phi} \phi_i \frac{1}{r_2^*} c_i b P_0^{(1)})^{-1} \cdot \\
&\qquad \hat{e}(r_2^* P_0^{(2)}, \sum_{i\in\Phi} \phi_i \frac{1}{r_2^*} a_i c P_0^{(1)})^{-1} = \hat{e}(P_0^{(1)}, P_0^{(1)})^{2(abc)} \cdot \hat{e}(aP_0^{(1)}, bP_0^{(1)})^{-1}
\end{aligned}$$

When $\mathcal{A}$ wins the game with non-negligible probability, then it suffices that $\mathcal{B}$ picks the right entry $(\alpha, h_{2,\alpha})$ of $L_{\mathcal{H}_2^{sim}}$, in order to solve the BDH (by computing $(\alpha \cdot \hat{e}(aP_0^{(1)}, bP_0^{(1)}))^{\frac{1}{2}}$). Consequently, $\mathcal{B}$ solves the BDH if and only if $\mathcal{A}$ computes $\mathcal{H}_2(\eta)$ and $\mathcal{B}$ picks the right entry of $L_{\mathcal{H}_2^{sim}}$. Let $\varepsilon(k)$ be $\mathcal{A}$'s advantage in the *IND-thABE-CPA* game. Then, $\mathcal{B}$ solves the BDH problem with probability at least $\frac{\varepsilon(k)}{q_{\mathcal{H}_2}}$, where

$q_{\mathcal{H}_2}$ is the number of $\mathcal{H}_2$ queries issued by $\mathcal{A}$ in the attack game. Moreover, since each query of the *IND-thABE-CPA* game requires $\mathcal{B}$ to make a polynomial number of operations in $\mathbb{Z}_q^*$, $\mathbb{G}_2$ and $\mathbb{G}_1$, it follows that $\mathcal{B}$ solves the BDH problem in time $O(\tau)$ where τ is the running time of $\mathcal{A}$ in the *IND-thABE-CPA* attack game. **QED**.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005
© 2005 The authors. All rights reserved.

A Study on a Framework of Online Biometric Authentication with Verification of Personal Repository

Yoshifumi UESHIGE[a] and Kouichi SAKURAI[a,b]

[a] *Institute of Systems & Information Technologies/KYUSHU, Fukuoka SRP Center Building 7F, 2-1-22 Momochihama, Sawara-ku, Fukuoka, 814-0001, Japan*
ueshige@isit.or.jp, sakurai@isit.or.jp
[b] *Department of Computer Science and Communication Engineering, Kyushu University, 6-10-1 Hakozaki, Fukuoka, 812-8581, Japan*
sakurai@csce.kyushu-u.ac.jp

Abstract. Biometric authentication is remarkable with respect to identification of legitimate users. Biometric authentication is hopeful of services on the internet as reinforcement for conventional authentication such as ID and password, however, biometric information –acquisition raw data and template data– is unrenewable even though the data is compromised. We propose a framework of online biometric authentication with verification of validity of user's personal repository based on PKI. In this framework, information of biometrics authentication (certificate of templates) is related to not information of ownership but personal repository. This framework achieves anonymity during biometric authentication process by verifying validity of the user's personal repository.

Keywords biometrics, personal repository, certificate, Bio-PKI, framework

1. Introduction

1.1. Background

Today, authentication techniques are significant in various IT services. The authentication techniques can be classified into three categories, that is, i) the authentication based on user's knowledge (e.g. ID-passwords, and PINs), ii) the authentication by using user's possessions (e.g. magnetic cards, and IC cards), iii) biometric authentication (e.g. fingerprint, luster, vein layout, facial images and etc.).

Some techniques cannot prevent illegalities because of vulnerability as shown in forgery of credit cards and cash cards. In many authentication schemes, biometrics authentication is expected as one of remarkable and hopeful solution [3], however, cost of devices, and enrollment is needed.

Many internet services (E-commerce, internet banking, and etc.) require strong user identification and remote authentication techniques. As infrastructure of authentication on open networks like the internet, public key infrastructure (PKI) [2] is well known. In PKI, a user generates a pair of keys which are private key and public key. The user enrolls the pair of keys to certificate authority (CA). By verifying a public key certificate (PKC) which CA issues, service providers can authenticate user who has valid pair of keys. Because the authentication scheme in PKI is based on user’s possessions, authentication issue of losing possessions is not solved sufficiently.

About this problem, some works of biometric authentication on the open networks have proposed [4] ~ [9]. They are called BioPKI. BioPKI is a framework for user authentication with not only PKC but also biometrics. Isobe and Seto [4] proposed a public-key infrastructure (PKI) with biometric authentication using template formats. Their framework is additional to PKI as a substitution for qualified certificates. Ikeda, Morijiri, and Saisho [5] proposed a framework of verification of biometrics authentication environments and authentication results. These two proposals are similar with roles of entities in their frameworks, however, treatment of biometric data or templates are different between two proposals. In [4], CBEFF [6] was applied to format of template data and digital signature of the template data was added to the data in order to transmit the template data. This format includes information of corresponding public-key certificate. Therefore, this framework has possibility of eavesdropping biometric data and its owner’s privacy. On the other hand, in [5] biometric authentication was done in clients and the authentication results were transmitted to application server. Hence the biometric data or the templates were not sent. The framework can be applied to only client matching model.

1.2. Our contribution

We propose a framework of online biometric authentication with verification of validity of user's personal repository. In this framework, information of biometrics authentication (ex. certificate of templates) is related to not information of ownership but personal repository. Certificate of the personal repository is issued and distributed restrictedly. This certificate is verified with respect to validity of the personal repository by verification authority (VA). In this framework, the entities which can obtain the certificate of the personal repository are owner, CA and VA of the personal repository. We think this causes privacy protection. Our proposal provides the framework which achieves anonymity during biometric authentication by verifying validity of the user's personal repository.

In section 2, we describes summary of [4], [5] as related works. We explain our proposal and discuss security in section 3. Finally, we conclude this paper in section 4.

2. Related works

2.1. PKI with biometric authentication using template format [4]

This work is proposal that template format which is independent of public-key certificate is defined because of absorption of difference in biometrics authentication algorithms, accuracy of template data, and life time of template data. Isobe and Seto states that PKI with biometric authentication is improved by introducing the template format.

Table1 Biometric template format

#	Item	Contents
1	Format Identifier	Identification information of template format
2	Certificate Identity Information	Identification information of corresponding public key certificate. (ex. issuer's name & serial number of the certificate)
3	Issuer Name & ID	Issuer's name who issued this template format and his/her identification information
4	Template Data	Biometrics type, template data, information of authentication algorithms and so on (based on CBEFF)
5	Issuer's Digital Signature	Digital signature for data of #1 to #4 including digital signature scheme

2.1.1. Entities

This framework requires following authorities for biometric authentication.

i) Certificate authority (CA) of public-key,

ii) Biometrics issuing authority (BIA) for registering and issuing template formats,

iii) CA of evaluated results of threshold and accuracy for biometrics techniques,

iv) CA with respect to evaluated results of biometrics device security.

2.1.2. *Definition of template format*

Table1 shows template format in [4]. This format has a field "**Certificate Identity Information** ". Generally, public-key certificates include information of user's name and some privacy. Hence, anyone can easily find a pair of this template format and its owner's (user) information. **Template data** is a field of CBEFF data [10] which includes template data with or without encryption. As CBEFF include various form of biometrics templates themselves, there is possibility that adversaries steal the CBEFF data in some cases. It is able for adversaries to apply this table to pretend the owner.

2.1.3. *Authentication process using the template format*

Flaw of authentication process is shown in Figure 1. Roughly speaking, this process consists of two phases. First phase is checking integrity of public-key certificate and template format. Second is biometric matching on biometrics device.

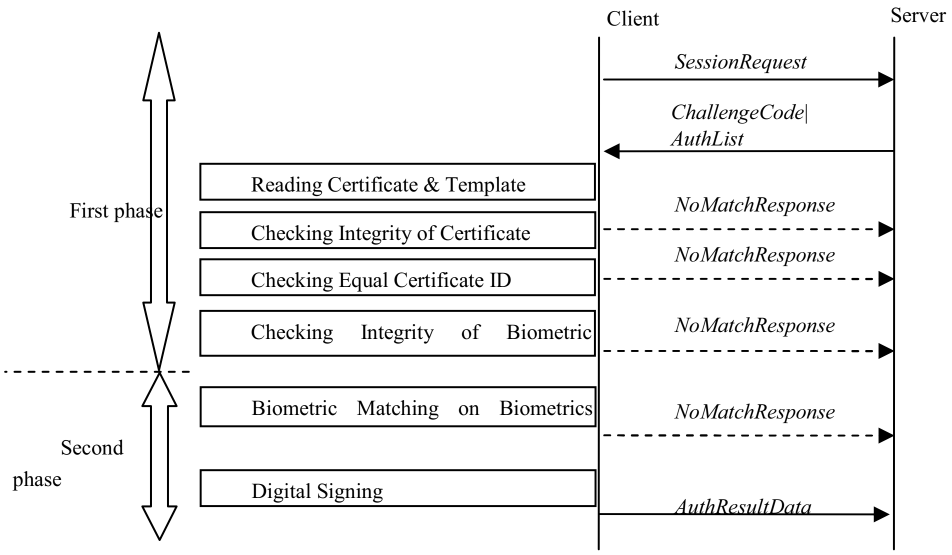


Figure 1 Authentication process by using biometric template format

2.2. *Biometric authentication by verifying environments and authentication results[5]*

In this framework, application sever can verify biometric authentication results by checking validity of authentication environments and authentication results. This

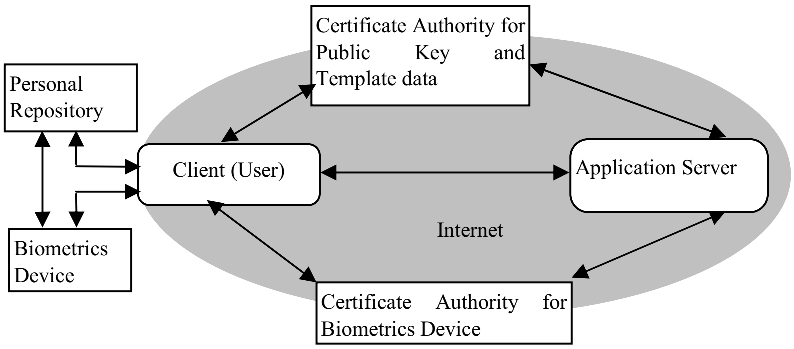

Certificate Authority for Public Key and Template data
Personal Repository
Client (User)
Application Server
Internet
Biometrics Device
Certificate Authority for Biometrics Device

Figure 2 Framework of biometric authentication

framework is applied to only client authentication model, because biometric information (raw data, templates, etc.) is not communicated on the internet.

2.2.1. *Structure of biometric authentication framework*

The structure of biometric authentication *framework* is shown in Figure 2. This framework consists of following entities.

i) Client has biometric devices which is tamper resistant. Acquisition of biometric raw data and matching are done on the biometric devices.

ii) Users have own personal repositories which are tamper resistant devices like smart cards. Qualified certificate [11] and private key are stored in the repositories.

iii) CA of public-key and biometric templates issues qualified certificates.

iv) CA of authentication environment issues certificates of security of biometric devices, security and accuracy of matching algorithms.

2.2.2. *Authentication with authentication environment verification*

Authentication protocol consists of following three steps:

i) Negotiation for starting authentication process

For authentication request from the client ***SessionRequest***, session information ***SessionInfo***, public-key certificate of the application server $\boldsymbol{cert_S}$, and a list of available authentication types ***AuthList*** are sent from the application server.

ii) Biometrics authentication

In communication between the personal repository and the biometric device, process of biometric authentication is achieved. First, the personal repository and the biometric device are mutually authenticated with challenge-response method. The mutual authentication is implemented with session key ***sk*** from personal repository

Next, the personal repository sends encrypted templates by ***sk*** $\mathbf{Enc}_{sk}(\boldsymbol{Template})$ to biometric device. The biometric device checks biometric raw data with the templates. The biometric device returns verification result data ***VerfResultData***.

iii) Proposition of authentication results

Final result ***AuthResultData*** that is a set of the verification result data ***VerfResultData***, user's public-key certificate $\boldsymbol{cert_U}$, and device certificate of biometric device $\boldsymbol{cert_{BD}}$ are transmitted to the application server throughout the client. The application server checks the final result.

3. Proposed framework of remote biometric authentication

We propose a framework which authenticates legitimate user by using biometrics authentication in online case. In our proposal, we regard set of certificates of public keys, and one of biometric template as personal information, because anyone can obtain personal information (user's name and his/her biometric templates) from these certificates. We provide the framework where these certificates cannot be used easily by any adversaries. Following sections points out problems of the above conventional methods, and describes our solutions.

3.1. Remarkable points

Above two proposals have similar points in their frameworks. In particular, entities of the framework and their authentication process seem to be common. On the other hands, differences between the proposals are following points. Ref [5] assumes legitimate users possess personal data repositories like smart cards. The personal repositories require capability of verification for certificate. The framework has a problem of biometric data protection. That is, qualified certificate is used as certificate of biometric information in this framework. Qualified certificates can be obtained easily by anyone, because the qualified certificates are distributed from web server. The qualified certificates are not proper for security of biometric templates. Therefore, we think that CA is required for the certificate distribution rather than web server.

In ref [4], BIA require more strict security management of the template format than one in CA of public-key certificate, because the template format includes the template data directly. For this problem, we modify the biometric template format such that CA of template data can manage certificate of biometric templates and the related template data, separately.

Biometrics information must be prevented from compromising and misuse, because the biometric information is essentially irreplaceable. Therefore, all entities must assure trustees each other by negotiating and authenticating mutually before biometric authentication process. In our solution, in order not to compromise privacy information from entities the user and the application server verify the biometric authentication environment before the authentication process.

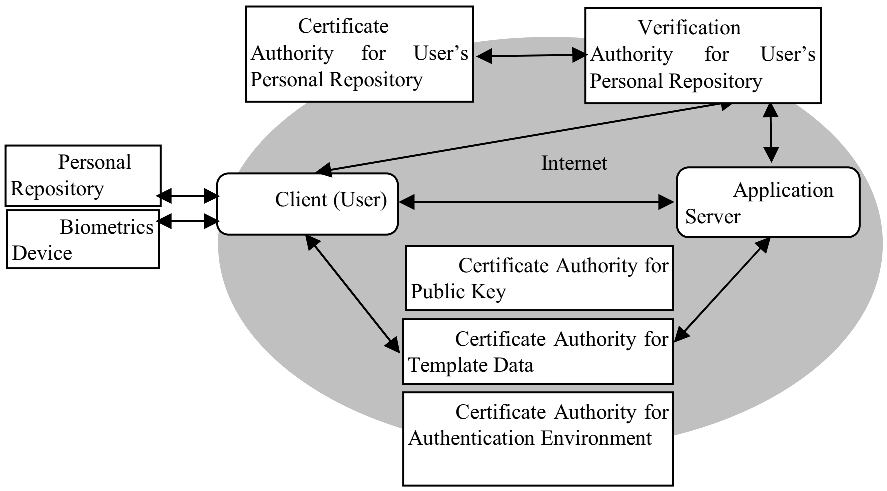

Certificate Authority for User's Personal Repository
Verification Authority for User's Personal Repository
Personal Repository
Biometrics Device
Internet
Client (User)
Application Server
Certificate Authority for Public Key
Certificate Authority for Template Data
Certificate Authority for Authentication Environment

Figure 3 Proposed framework of online biometric authentication

3.2. Overview of our framework

We show a proposal framework in Figure 3. This framework contains the following entities.

- **User** who has a personal repository stores private keys, certificate of public keys, certificate of biometric template, certificates of the personal repository, registered biometric templates data.
- **Application server** which provides online services after biometric authentication.

- **Certificate authority (CA) of user's personal repository** which distributes the certificate restrictedly.
- **Verification authority (VA) of user's personal repository** which verifies validity of personal repository and whether the corresponding personal repository stores the certificates which the application server is required or not.
- **CA of public-keys** which distributes public-key certificates.
- **CA of biometric template** which distributes certificates of biometric template that implies hash value of biometric templates for verification of the provided biometric templates data.

3.3. *Definition of certificates of user's personal repository*

The certificate of personal repository is defined as table 2. This certificate relates the public-key certificates and biometric template certificates by the field "**Owner's**

Table 2 Certificates of user's personal repository

#	Item	Contents
1	Identify Information of personal repository.	Identification information of personal repository. (ex. Vender information, serial number of the personal repository)
2	Owner's Identity Information	Identification information of owner of personal repository.
3	Issuer Name & ID	Issuer's name who issued this template format and his/her identification information
4	Validity	Life time of the certificate
5	The number of stored certificates, ***n***	The number of certificates which are stored in the personal repository
6	Information of stored certificate 1	Information of the certificate No. 1 stored in the personal repository (ex. Type of certificate, address of CA, and ID of the certificate)
	…	…
	Information of stored certificate ***n***	Information of the certificate No. ***n*** stored in the personal repository
	Issuer's Digital Signature	Digital signature for above data including digital signature scheme

Identity Information", because the biometric template certificate includes not user's identification information but identification information of his/her personal repository. This purpose of the certificate is that eavesdropping privacy information by obtaining public-key certificate and biometric template certificate simultaneously is prevented.

Also this certificate has a list of certificates stored in the personal repository. From this list, the application server can verify applicable authentication schemes before performing authentication process. The CA of personal repository distributes this certificate to only the VA; therefore, the other entities cannot know the relationship between the personal repository and its owner.

3.4. Definition of biometric template certificate

In table1, the template format explicitly includes the fields of "**Certificate Identity Information**", and "**Template Data**". Because this format has problems pointed out in section 3.1, we modify the template format to biometric template certificate shown in table 2 in consideration for privacy protection. The template certificate implies a field "**Personal Repository Identity Information**" for relating with personal repository. Also, since hash value of biometric templates is included, risk of compromising biometric templates themselves is reduced.

Table 3 Proposed biometric template certificate

#	Item	Contents
1	Certificate Identifier	Identification information of template certificate Ex. serial number, life time of the certificate.
2	Personal Repository Identity Information	Identification information of user's personal repository Ex. IP address, identification information of device certificate
3	Issuer Name & ID	Issuer's name who issued this template format and his/her identification information
4	Template Information	Biometrics type, hash value of template data, information of authentication algorithms and so on
5	Issuer's Digital Signature	Digital signature for data of #1 to #4 including digital signature scheme

3.5. Authentication process

This subsection describes authentication process. The authentication process consists of two steps. First step is verification process for legitimating user's personal repository. Second step is biometric authentication process. We make these processes more precise.

1) verification process for legitimacy of personal repository

Following processes verify validity of the personal repository based on its certificate. The flow of these processes is shown in figure 4. These processes can achieve anonymity for the application server.

i) User sends request of starting authentication session ***RequestAuthSession***

ii) Application server sends session starting information ***SessionInfo*** as a response.

iii) The user sends device certificate of personal repository $\boldsymbol{Cert_{PRdevice}}$ to application server. The user does not send certificate of ownership of his/her personal repository, because of ensuring anonymity in this step.

iv) The application server sends request $\boldsymbol{RequestVerf_{Prowner}}$ to the VA for verifying legitimate owner of the personal repository exists and to ensure the

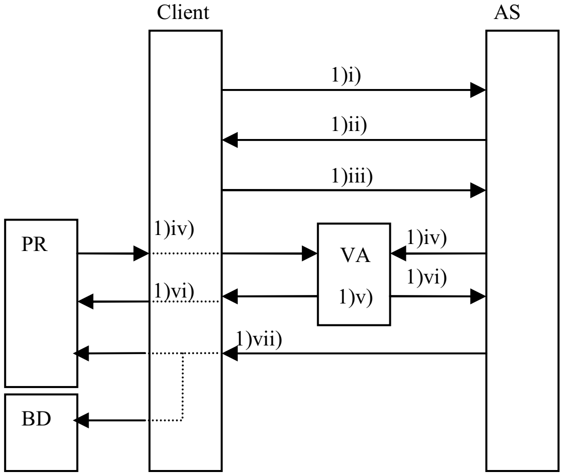


Figure4 Authentication Process for Personal Repository

(AS: Application Server, PR: Personal Repository, BD: Biometric Devices, VA: Verification Authority) user has certificates which the application server needs to authenticate with sending the identification information of personal repository and information of available authentication method on the application server. Simultaneously, the user sends the same request $\boldsymbol{RequestVerf_{Prowner}}$ to the VA with sending own certificate of ownership of personal repository $\boldsymbol{Cert_{Prowner}}$, and the device certificate $\boldsymbol{Cert_{PRdevice}}$. Both requests are necessary to confirm an agreement of authentication process between the user and the application server.

v) After ensuring above two requests, the VA implements verification process. The VA verifies the ownership certificate $\boldsymbol{Cert_{Prowner}}$ and the device certificate $\boldsymbol{Cert_{PRdevice}}$ of the personal repository from verifying the digital signatures of the certificates, and revocation list of each certificate. The VA generates following information as the verification results and its digital signature.

$$\boldsymbol{VerfResult = OwnerExist | CertExist | \mathrm{enc}_{kVA}(\mathrm{h}(OwnerExist \mid CertExist))},$$

where ***OwnerExist, CertExist*** denote information about existence of valid owner, and certificates which application server requires, respectively. Also ***kVA*** denotes private key of VA.

vi) The VA sends ***VerfResult*** to the application server.

vii) The application server verifies ***VerfResult***. If there is the valid owner of the personal repository and the user (i.e. the owner) has certificates required for authentication, the application server sends a list of biometrics which the application server can perform ***BioAuthList*** to the user. If else, the application server sends ***ExitSession***, and biometric authentication process ends.

2) Biometric authentication process

This process consists of selection of biometric devices based on ***BioAuthList***, verification of biometric authentication environment, verification of template certificate, matching and decision process. The flow of these processes is shown in figure 5. The feature of this process, information including user's name does not be sent, therefore, no one can obtain set of user's name and his/her biometric information. Specification of this process is as follows.

i) The user collects device certificates of biometric devices $\boldsymbol{Cert_{BD}}$ connected with the client based on ***BioAuthList***.

ii) After collecting device certificates in the user's client, the user sends the certificates $\boldsymbol{Cert_{BD}}$ with digital signature by the personal repository to the application server. If there is lack of biometric devices on the client, the client sends ***NoBioAuthSys***.

iii) The application server verifies the device certificates. When environment of biometric authentication in the client are valid and satisfies security policy, the application server sends biometric authentication starting code ***StartBioAuth***. If the environment does not satisfy this condition, the application server exits the biometric authentication after sending ***ExitSession***.

iv) The user sends the template certificates such as table 3 to the biometric devices.

v) The biometric devices verify the template certificates. When the corresponding templates have sufficient quality, the biometric devices request the templates to the personal repository.

vi) The user sends the templates to the biometric devices.

vii) The biometric devices perform matching process and decision process. The biometric devices send the result of the authentication to the application server.

viii) The application server verifies the template certificate, and checks the

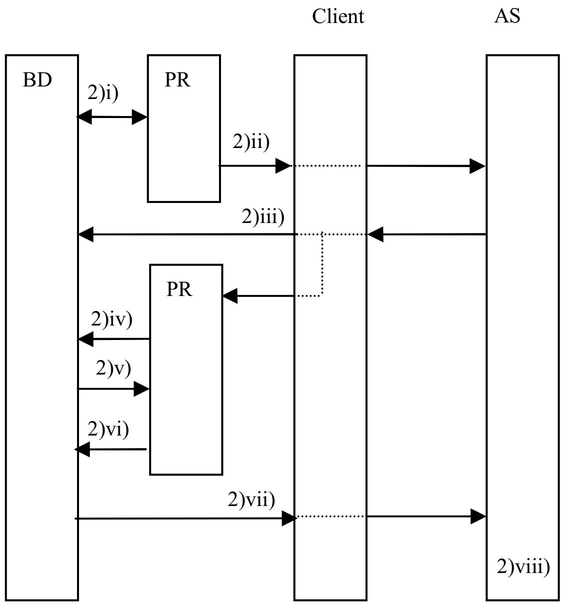


Figure 5 Biometric Authentication Process

quality of biometric template from the template certificate. If verification is succeeded, and the user's biometric template data has enough quality, the biometric authentication process is performed according to the biometric authentication environment between the user and the application server.

3.6. Security Requirement

Our proposal is the framework which performs verification of ownership of the corresponding personal repository by using biometric authentication. In the step of the biometric authentication, anonymity is assured. The VA can verify the relationship between the user and the owner of the personal repository. Therefore, while the VA is credible, this framework is secure from the viewpoint of privacy protection. On the other hand, because the result of the verification is presence of the valid owner for the personal repository which is used in authentication process, if the VA and malicious user conspire, this framework cannot provide secure environment.

In order to prevent alteration of communication data, personal repository should have calculation power for generating and verifying digital signatures.

4. Conclusion

In this paper we have proposed the framework of biometric authentication on open network. This proposal has effect of ensuring anonymity of the user until biometrics authentication process is accomplished because whole of user is distinguished by identify information of personal repository only. Also, an effect of privacy protection follows the effect of anonymity in the authentication. Finally, we discussed security requirement of the framework.

Future works are as follows. First is extension of the framework to more general authentication model. Second is treatment of biometric information in order not to abuse it remaining in the devices.

Acknowledgment

A Part of this research was supported by TAF (The Telecommunication Advancement Foundation).

References

[1] Russell Housley, Warwick Ford, Tim Polk, David Solo, “Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile”, RFC3280, Network Working Group Standard Track, (2002),

URL: http://www.faqs.org/rfcs/rfc3280.html

[2] Antonio Lioy, Marius Marian, Natalia Moltchanova, *et al.* ,”The EuroPKI Experience”, Public Key Infrastructure: First European PKI Workshop: Research and Applications, EuroPKI 2004 (LNCS3093), pp. 14 – 27, (2004),

URL: http://security.polito.it/doc/papers/EuroPKI2004.pdf

[3] Paul Reid, “Biometrics for Network Security”, PRENTICE HALL Professional Technical Reference, (2004)

[4] Yoshiaki Isobe, Youich Seto, Masanori Kataoka, “Development of Personal Authentication System Using Fingerprint with Digital Signature”, Proceedings of the 34th Hawaii International Conference on System Sciences 2001 (2001)

[5] Tatsuro Ikeda, Tomoaki Morijiri, Toshiaki Saisho, “A Proposal of Authentication Scheme for User-Identification Environment”, Proceedings of Computer Security Symposium 2004, pp. 337-342, (2004)

[6] Yasushi Yamazaki, Naohisa Komatsu, “A Secure Communication System Using Biometric Identity Verification”, IEICE Transaction of Information & Systems, Vol. E84-D, No. 7, pp. 879-884, 2001

[7] Kaoru Uchida, “Fingerprint Identification for Enhanced User Interface and for Secure Internet Services”, IEICE Transaction of Information & Systems, Vol. E84-D, No. 7, pp. 806-811, 2001

[8] Hao Feng, Chan Choong Wah, “Private key generation from on-line handwritten signatures”, Information Management & Computer Security, 10/4, pp.159-164, 2002

URL:http://www.cl.cam.ac.uk/users/fh240/pdf/Private%20key%20generation%20from%20 on-line%20handwritten%20signatures.pdf

[9] Yukio Itakura, Shigeo Tsujii, “Proposal on Bio-PKI in which DNA Personal Identifier is embedded in Public Key”, The Third International Workshop for Applied Public Key Infrastructure, pp. 108-113, Oct. 2004.

[10] Fernaldo L. Podio, Jeffrey S. Dunn, Lawrence Reinert, Catherine Tilton, Lawrence O’ Gorman, M. Paul Collier, Mark Jerde, Brigitte Wirtz, “Common Biometric Exchange File Format (CBEFF)”, National Institute of Standards and Technology (NIST), (2001)

URL: http://www.itl.nist.gov/div895/isis/bc/cbeff/CBEFF010301web.PDF

[11] Stefan Santesson, Tim Polk, Magnus Nystrom, “Internet X. 509 Public Key Infrastructure: Qualified Certificate Profile”, RFC3739, Network Working Group Standard Track, (2001)

URL: http://www.faqs.org/rfcs/rfc3739.html

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005
© 2005 The authors. All rights reserved.

On Universal Composable Security of Time-Stamping Protocols

Toshihiko Matsuo [a,1], Shin'ichiro Matsuo [a,2]
[a] *NTT DATA Corporation*

Abstract. Time-stamping protocols, which assure that a document was existed at a certain time, are applied to some useful and practical applications such as electronic patent applications and so on. There are two major time-stamping protocols, the simple protocol and the linking protocol. In the former, a time-stamp authority issues a time-stamp token that is the digital signature of the concatenated value of a hashed message and the present time. In the latter, the time-stamp authority issues a time-stamp token that is the hash value of the concatenated value of a hashed message and the previous hash value. Although security requirements and analysis for above time-stamping protocols has been discussed, there are no strict cryptographic security notions for them. In this paper, we reconsider the security requirements for time-stamping protocols and define security notions for them, in a universally composable security sense, which was proposed by Canetti. We also show that these notions can be achieved using combinations of a secure key exchange protocol, a secure symmetric encryption scheme, and a secure digital signature scheme.

Keywords. time-stamping protocol, universal composable security

1. Introduction

Opportunities for creating a lot of digital documents and distributing them over digital networks are growing rapidly. They include not only various kinds of private documents but also important documents such as formal applications and business contracts. It is thus important to be able to prove that a digital document existed at a certain time. For example, contracts are dated and the validity of the date may later have to be proven. For patents, the exact date when it was applied must be shown, and it must be possible to prove the validity of the date. In stock trading, the exact time when a buy/sell order was placed must be shown since the gain/loss is directly affected by the time.

Two major time-stamping protocols, the simple protocol and the linking protocol [ISO,HS91,ACPZ01] that satisfy these requirements have been developed. The former uses a digital signature as a time-stamp token [ISO,ACPZ01]. A time stamp authority (TSA) signs the concatenation of the hashed message and the present time. by using its private signing key. Since the validity of the time-stamp token depends on that of the digital signature scheme itself, the TSA must be a trusted third party. In the latter, a

[1]E-mail: matsuotsh@nttdata.co.jp
[2]E-mail: matsuosn@nttdata.co.jp

time-stamp token is the hash value of the concatenation of the present hashed message and the previous hash value [HS91]. A verifier can check the validity of the token by using published values for the hash chain. In this case, the TSA is not necessarily a trusted third party. Several variants of linking protocols have been proposed such as [BLLV98,BLS00]. These two major protocols are already being used by actual time-stamping services such as Surety [Sure].

1.1. Security requirements of time-stamping protocols

There are two major types of attacks on time-stamping protocols. The first type is that an adversary may try to back-date the valid time-stamp. This is a fatal attack for applications in which the priority is based on descendent time order. [1] This type of attack is called a "*back-dating attack*." The adversary may corrupt the TSA and may try to create a forged but valid time-stamp token. The simple protocol is clearly not secure against TSA corruption, but the linking protocol is since a verifier can check the validity by computing the chain of hash values using published hash values.

In the other type of attack, an adversary may try to forward-date the time-stamp without the approval of the valid requester. This is a fatal attack for applications in which the priority is based on ascendent time order. [2] This type of attack is called a "*forward-dating attack*."

Although there have been several studies of the security of time-stamping protocols [HS91,Just98,UM02], there are no strict security notions for them in a cryptographic (computational) sense.

1.2. Universally composable security

Canetti proposed a framework for defining the security of cryptographic protocols that he called *universally composable security* (UC security) [C01]. In this framework, the ideal functionality that achieves a certain service, the set of parties and the adversary are denoted as $\mathcal{F}$, $\widetilde{P}$, and $\mathcal{S}$, respectively. Each party does not communicate directly with the others, and the adversary can corrupt any party at any time. On the other hand, the actual protocol that achieves the service, the set of parties, and the adversary are denoted as π, P, and $\mathcal{A}$, respectively. Then, we assume the existence of an environment $\mathcal{Z}$ which communicates all parties and $\mathcal{A}$. Each party can communicate with the others and $\mathcal{A}$ can control all communication, meaning that $\mathcal{A}$ can read or alter all messages among the parties. $\mathcal{A}$ can also corrupt any party at any time. In Canetti's framework, protocol π securely realizes $\mathcal{F}$ if, for $\forall\mathcal{A}$ and $\forall\mathcal{Z}$, there exists an adversary $\mathcal{S}$ which makes $\mathcal{Z}$ difficult to distinguish whether she accesses $\widetilde{P}$ and $\mathcal{S}$ or P and $\mathcal{A}$. This framework helps us to prove security of large cryptographic protocol due to following two properties

Composition Theorem: The key advantage of UC security is that we can create a complex protocol from already-designed sub-protocols that securely achieve the given local tasks. This is very important since complex systems are usually divided into several sub-systems, each one performing a specific task securely. Canetti presented this feature as the composition theorem [C01]. This theorem assures that

[1] For example, intellectual property rights protection is the case.

[2] For example, digital will is the case.

we can generally construct a large size "UC-secure" cryptographic protocols by using sub-protocols which is proven as secure in UC-secure manner.

Hybrid model: In order to state above theorem and to formalize the notion of an actual protocol with access to multiple copies of an ideal functionality, Canetti also introduced the hybrid model which is identical to the actual model with the following. On top of sending messages to each other, the parties may send messages to and receive messages from an unbounded number of copies of an ideal functionality $\mathcal{F}$. The copies of $\mathcal{F}$ are differentiated using their session identifier SIDs. All messages addressed to each copy and all messages sent by each copy carry the corresponding SID.

There are various studies on the sense of UC security. Although several ideal functionalities of cryptographic primitives have been proposed [C01,CF01,CK02,C04] [3], there is no definition of functionality of time-stamping protocol.

1.3. Our contribution

In this paper, we consider the security notions of the time-stamping protocol and define its functionality based on the UC framework. Our definition follows that of the signature functionality, $\mathcal{F}_{SIG}$, defined by Canetti [C04] because the required properties of the time-stamping protocol are similar to those of the digital signature scheme. However, a time-stamping protocol requires unique security properties, so we have to newly define its functionality.

In addition, we describe the construction of a secure time-stamping protocol π_{TS} using the key exchange functionality, $\mathcal{F}_{KE}$, [C01] and the signature functionality. The construction of this protocol is similar to the simple protocol. Briefly speaking, a time-stamp token requester and a TSA exchange a session key by using $\mathcal{F}_{KE}$, and then the requester encrypts the message by using the session key and sends it to the TSA. Then, the TSA time-stamps the received message by using $\mathcal{F}_{SIG}$ so as to include the requester's ID and returns the token to the requester. Since the message the requester wants to be stamped is encrypted, an adversary can not obtain a time-stamp token ahead of a valid requester. The requester's ID prevents the adversary from claiming her legitimacy with a time-stamp token she acquires by observing the transaction.

Organization of this paper is as follows. In section 2, we give our definition of time-stamping protocols, and describe their security requirements and ideal functionality. Then we show the construction of a UC secure time-stamping protocol and its security proof in section 3. In section 4, we discuss a simpler construction and its security. We conclude our study in section 5 with a brief summary.

2. Time-Stamping Protocols

2.1. Definition

In this paper, we define a time-stamping protocol as follows.

[3] Digital signature, public-key encryption, key exchange, bit commitment etc.

- Let k be a security parameter. A TSA obtains time-stamping key δ and verification key θ by executing key generation protocol or algorithm $SetUp$, which outputs δ and θ on input 1^k.
- Each time-stamp token requester executes a time-stamp token generation protocol and acquires the time-stamp token σ for document d and time t from the TSA.
- A verifier verifies σ by executing time-stamp token verification protocol or verification algorithm Ver. He verifies σ with θ and auxiliary information ρ.

In the simple protocol δ and θ are TSA's signing key and verification key of underlying digital signature scheme. In the linking protocol, δ and θ are selected hash function of hash chain and ρ contains hash values for time-stamp token verification. In the following, we denote a time-stamping protocol by π_{TS}.

2.2. Security requirements

We let an adversary for a time-stamping protocol be an interactive Turing machine (ITM) [4] [C04]. The adversary (we sometimes denote an adversary by $\mathcal{A}$) can do anything to the communication between any two parties. $\mathcal{A}$ can also corrupt any party at anytime. The security requirements for a time-stamping protocol are similar to those for a digital signature; however, we have to take the following requirements into consideration.

1. $\mathcal{A}$ may initiate a man-in-the-middle attack because a time-stamp requester can not issue a time-stamp token by herself; the requester has to communicate with a TSA.
2. To protect "forward-dating attack," the time-stamp token should contain the requester's ID to make the protocol secure [MO04].
3. In the linking protocol, the verification algorithm needs not only a verification key but published hash values to enable σ to be verified.

Therefore, we define the security requirements for a time-stamping protocol as follows.

Definition 1 *Let k be a security parameter and $\varepsilon(\cdot)$ be a negligible function on k. Let δ be a time-stamping key, θ be a verification key, and ID be a unique identifier of the requester. We say that a time-stamping protocol satisfies the security requirements if the following properties hold.*

Completeness *For any document d and valid time-stamp token σ stamped at time t,*

$$\Pr_{\delta}[(\delta,\theta) \leftarrow SetUP(1^k); 0 \leftarrow Ver(\theta,\rho,t,d,ID,\sigma)] \leq \varepsilon(k),$$

where ρ is valid auxiliary information generated during execution of the protocol.

Consistency *For any document d and valid time-stamp token σ stamped at time t, the probability that $Ver(\theta,\rho,t,d,ID,\sigma)$ generates two different outputs in two independent invocations is smaller than $\varepsilon(k)$, where ρ is valid auxiliary information generated during execution of the protocol.*

[4] We sometimes denote an ITM entity by using a calligraphic font.

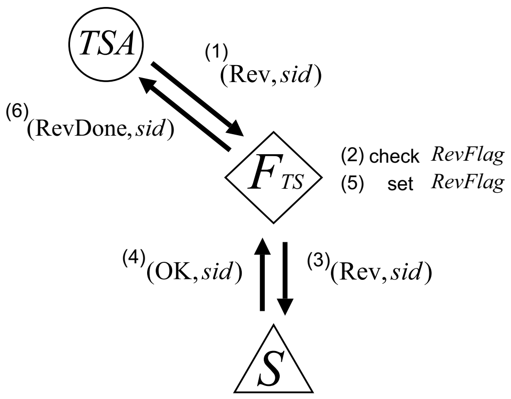


Figure 1. Key Revocation process

Unforgeability $\Pr_{\delta}[(\delta,\theta) \leftarrow SetUP(1^k); \{(t_0, d_0, ID_0, \sigma_0, \rho_0), (t_1, d_1, ID_1, \sigma_1, \rho_1)\} \leftarrow \mathcal{A}^{\pi_{TS}}(\theta); b \in \{0,1\}; 1 \leftarrow Ver(\theta, \rho_b, t_b, d_b, ID_b, \sigma_b)] \leq \varepsilon(k)$,

where ρ_b is valid auxiliary information generated during execution of the protocol. Furthermore, (1) either σ_0 or σ_1 is not generated by TSA or (2) $d_0 = d_1$, $ID_0 = ID_1$, and $t_0 \neq t_1$.

2.3. Ideal functionality and security condition

In the UC framework, all entities are interactive Turing machines [C04]. Each entity has a session-identifier (SID) that represents the session to which the entity belongs. It also has a party identifier (PID) that represents the role of the entity in the protocol instance. The pair $sid = (PID, SID)$ is guaranteed to be unique in the system. We define the functionality $\mathcal{F}_{TS}$ of the time-stamping protocol as follows. For simplicity, we assume that a TSA manages only one time-stamping key at one time.

Key revocation (Fig. 1):

1. $\mathcal{TSA}$ sends $(\mathbf{Rev}, sid)$ to $\mathcal{F}_{TS}$.
2. If $sid = (\mathcal{TSA}, sid')$ for some sid' and the corresponding revocation flag $RevFlag$ equals to 1, go to the next step. Otherwise ignore the request.
3. $\mathcal{F}_{TS}$ sends $(\mathbf{Rev}, sid)$ to $\mathcal{S}$.
4. $\mathcal{S}$ sends $(\mathbf{OK}, sid)$ to $\mathcal{F}_{TS}$.
5. $\mathcal{F}_{TS}$ sets $RevFlag \leftarrow 0$, erases the corresponding record $(\mathcal{TSA}, \theta)$.
6. $\mathcal{F}_{TS}$ sends $(\mathbf{RevDone}, sid)$ to $\mathcal{TSA}$.

Key generation (Fig. 2):

1. $\mathcal{TSA}$ sends $(\mathbf{SetUp}, sid)$ to $\mathcal{F}_{TS}$.
2. If $sid = (\mathcal{TSA}, sid')$ for some sid' and $RevFlag = 0$, go to the next step. Otherwise ignore the request.

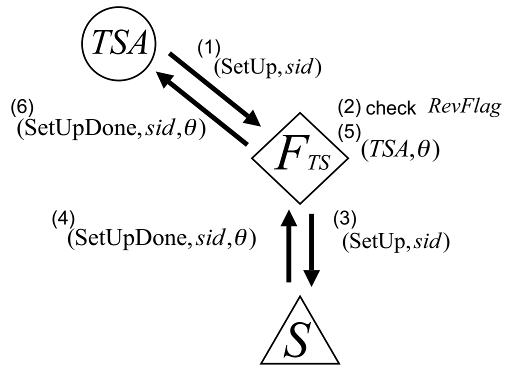


Figure 2. Key generation process

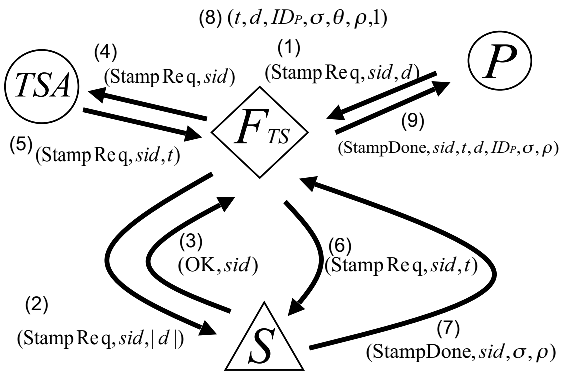


Figure 3. Time-stamp token generation process

3. $\mathcal{F}_{TS}$ sends $(\textbf{SetUp}, sid)$ to $\mathcal{S}$.
4. $\mathcal{S}$ sends $(\textbf{SetUpDone}, sid, \theta)$ to $\mathcal{F}_{TS}$.
5. $\mathcal{F}_{TS}$ records $(\mathcal{TSA}, \theta)$.
6. $\mathcal{F}_{TS}$ sends $(\textbf{SetUpDone}, sid, \theta)$ to $\mathcal{TSA}$. $\mathcal{F}_{TS}$ sets the corresponding flag $RevFlag \leftarrow 1$.

Time-stamp token generation (Fig. 3):

1. Time-stamp token requester $\mathcal{P}$ sends $(\textbf{StampReq}, sid, d)$ to $\mathcal{F}_{TS}$.
2. If $sid = (\mathcal{TSA}, sid')$ for some sid', $\mathcal{F}_{TS}$ sends $(\textbf{StampReq}, sid, |d|)$ to $\mathcal{S}$. Otherwise ignore the request.

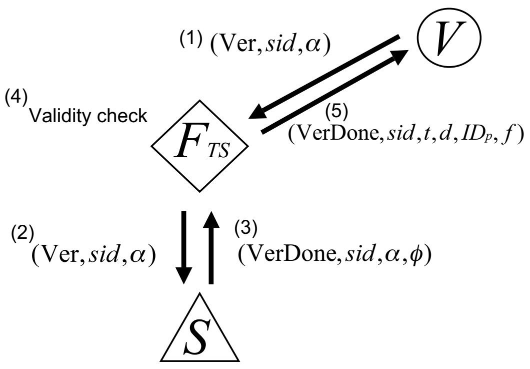


Figure 4. Time-stamp token verification process

3. $\mathcal{S}$ sends $(\mathbf{OK}, sid)$ to $\mathcal{F}_{TS}$.
4. $\mathcal{F}_{TS}$ sends $(\mathbf{StampReq}, sid)$ to $\mathcal{TSA}$.
5. $\mathcal{TSA}$ chooses t and then sends $(\mathbf{StampReq}, sid, t)$ to $\mathcal{F}_{TS}$, where t is an increasing value.
6. $\mathcal{F}_{TS}$ sends $(\mathbf{StampReq}, sid, t)$ to $\mathcal{S}$.
7. $\mathcal{S}$ sends $(\mathbf{StampDone}, sid, \sigma, \rho)$ to $\mathcal{F}_{TS}$.
8. $\mathcal{F}_{TS}$ records $(t, d, ID_{\mathcal{P}}, \sigma, \theta, \rho, 1)$.
9. $\mathcal{F}_{TS}$ sends $(\mathbf{StampDone}$, $sid, t, d, ID_{\mathcal{P}}, \sigma, \rho)$ to $\mathcal{P}$.

Time-stamp token verification (Fig. 4):
Let $f, \phi \in \{0, 1\}$.

1. Verifier $\mathcal{V}$ sends $(\mathbf{Ver}, sid, \alpha)$ to $\mathcal{F}_{TS}$, where $\alpha = (t, d, ID_{\mathcal{P}}, \sigma, \widetilde{\theta}, \widetilde{\rho})$.
2. $\mathcal{F}_{TS}$ sends $(\mathbf{Ver}, sid, \alpha)$ to $\mathcal{S}$.
3. $\mathcal{S}$ sends $(\mathbf{VerDone}, sid, \alpha, \phi)$ to $\mathcal{F}_{TS}$.
4. $\mathcal{F}_{TS}$ executes the following. (1) If $(\widetilde{\theta}, \widetilde{\rho}) = (\theta, \rho)$ and $\mathcal{F}_{TS}$ has already recorded $(\alpha, 1)$, $f \leftarrow 1$. (2) If $(\widetilde{\theta}, \widetilde{\rho}) = (\theta, \rho)$, $\mathcal{S}$ does not corrupt the $\mathcal{TSA}$, and $\mathcal{F}_{TS}$ has not recorded $(t, d, ID_{\mathcal{P}}, \sigma', \theta, \rho, 1)$ for $\forall \sigma'$, $f \leftarrow 0$ and $\mathcal{F}_{TS}$ records $(\alpha, 0)$. (3) If $(\widetilde{\theta}, \widetilde{\rho}) \neq (\theta, \rho)$ and $\mathcal{F}_{TS}$ has already recorded $(\alpha, \widetilde{f})$, $f \leftarrow \widetilde{f}$. (4) Otherwise, $f \leftarrow \phi$ and $\mathcal{F}_{TS}$ records (α, ϕ).
5. $\mathcal{F}_{TS}$ sends $(\mathbf{VerDone}, sid, t, d, ID_{\mathcal{P}}, f)$ to $\mathcal{V}$.

We use Canetti's definition of the signature functionality $\mathcal{F}_{SIG}$ [C04] to define $\mathcal{F}_{TS}$. In the key generation, we define that $\mathcal{S}$ can choose the verification key θ. This is because the security requirements in Def. 1 do not restrict the distribution of θ. Similarly, we define that $\mathcal{S}$ can choose time-stamp token σ and auxiliary information ρ in the time-stamp token generation.

In the token generation, if $\mathcal{S}$ obtains document d itself during protocol execution, it is clear that she can acquire its time-stamp token ahead of a valid time-stamp requester.

That is, as soon as $\mathcal{S}$ gets d, she can ask the TSA to issue the time-stamp token for d as a valid requester, ahead of the valid time-stamp requester. This is a fatal attack in applications like electronic patent applications. Therefore, d must be kept secret until protocol execution ends.

We define the UC security condition of a time-stamping protocol as follows.

Definition 2 *Let $\mathcal{F}_{TS}$ be an ideal time-stamping functionality, $\widetilde{P}$ be the set of dummy parties, and $\mathcal{S}$ be an ideal adversary with access to $\mathcal{F}_{TS}$. Let π_{TS} be the actual time-stamping protocol, P be the set of actual parties, $\mathcal{A}$ be the actual adversary with access to π_{TS}, and $\mathcal{Z}$ be an environment which communicates P and $\mathcal{A}$. If for any $\mathcal{A}$ and $\mathcal{Z}$, there exists $\mathcal{S}$ such that $\mathcal{Z}$ can not distinguish which entities she accesses, we say that π_{TS} securely realizes $\mathcal{F}_{TS}$.*

3. Construction of UC secure time-stamping protocol

Canetti proposed key exchange functionality $\mathcal{F}_{KE}$ and basic signature functionality $\mathcal{F}_{SIG}$ [C01,C04]. In this section, we describe a construction of UC-secure time-stamp protocol π_{TS} that is based on $\mathcal{F}_{KE}$ and $\mathcal{F}_{SIG}$ and similar to the simple protocol . For simplicity, we omit the key revocation procedure.

3.1. Preliminaries

We use $\mathcal{F}_{SIG}$ and $\mathcal{F}_{KE}$ as proposed by Canetti [C01,C04] in our construction. Canetti defined $\mathcal{F}_{SIG}$ as follows.

Key generation:

1. Signer $\mathcal{P}$ sends $(\mathbf{KeyGen}, sid)$ to $\mathcal{F}_{SIG}$.
2. $\mathcal{F}_{SIG}$ verifies that $sid \overset{?}{=} (\mathcal{P}, sid')$ for some sid'. If it does not hold , $\mathcal{F}_{SIG}$ ignores the request. Else, $\mathcal{F}_{SIG}$ sends $(\mathbf{KeyGen}, sid)$ to $\mathcal{S}$.
3. $\mathcal{S}$ sends $(\mathbf{VerificationKey}, sid, \theta)$ to $\mathcal{F}_{SIG}$.
4. $\mathcal{F}_{SIG}$ records $(\mathcal{P}, \theta)$ and then sends $(\mathbf{VerificationKey}, sid, \theta)$ to $\mathcal{P}$.

Signature generation:

1. $\mathcal{P}$ sends $(\mathbf{Sign}, sid, d)$ to $\mathcal{F}_{SIG}$.
2. $\mathcal{F}_{SIG}$ verifies that $sid \overset{?}{=} (\mathcal{P}, sid')$ for some sid'. If it does not hold , $\mathcal{F}_{SIG}$ ignores the request. Else, $\mathcal{F}_{SIG}$ sends $(\mathbf{Sign}, sid, d)$ to $\mathcal{S}$.
3. $\mathcal{S}$ sends $(\mathbf{Signature}, sid, d, \sigma)$ to $\mathcal{F}_{SIG}$.
4. $\mathcal{F}_{SIG}$ looks for the record $(d, \sigma, \theta, 0)$. If it is found, $\mathcal{F}_{SIG}$ sends an error message to $\mathcal{P}$ and halts. Else, $\mathcal{F}_{SIG}$ sends $(\mathbf{Signature}, sid, d, \sigma)$ to $\mathcal{P}$ and then records $(d, \sigma, \theta, 1)$.

Signature verification:

1. Verifier $\mathcal{V}$ sends $(\mathbf{Verify}, sid, d, \sigma, \widetilde{\theta})$ to $\mathcal{F}_{SIG}$.
2. $\mathcal{F}_{SIG}$ sends $(\mathbf{Verify}, sid, d, \sigma, \widetilde{\theta})$ to $\mathcal{S}$.
3. $\mathcal{S}$ sends $(\mathbf{Verified}, sid, d, \phi)$ to $\mathcal{F}_{SIG}$.
4. Upon receiving $(\mathbf{Verified}, sid, d, \phi)$, $\mathcal{F}_{SIG}$ works as follows.

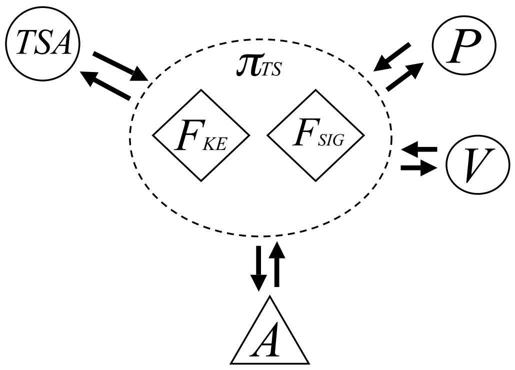


Figure 5. Hybrid construction of time-stamping protocol π_{TS}

1. If $\widetilde{\theta} = \theta$ and there exists the record $(d, \sigma, \theta, 1)$, $f \leftarrow 1$.
2. If $\widetilde{\theta} = \theta$, $\mathcal{P}$ has not yet been corrupted by $\mathcal{S}$, and there exists no record such that $(d, \widetilde{\sigma}, \theta, 1)$ for $\forall \widetilde{\sigma}$, $f \leftarrow 0$.
3. If $\widetilde{\theta} \neq \theta$ and there exists the record $(d, \sigma, \widetilde{\theta}, \widetilde{f})$, $f \leftarrow \widetilde{f}$.
4. Else, $f \leftarrow \phi$, then records $(d, \sigma, \widetilde{\theta}, \phi)$.

5. $\mathcal{F}_{SIG}$ sends $(\mathbf{Verified}, sid, d, f)$ to $\mathcal{V}$.

Canetti also defined $\mathcal{F}_{KE}$ as follows.

Functionality $\mathcal{F}_{KE}$:

1. Let $\mathcal{P}_i$ and $\mathcal{P}_j$ be two parties who want to share a key. $\mathcal{P}_i$ sends $(\mathbf{exchange}, sid, \mathcal{P}_i, \mathcal{P}_j, \beta)$ to $\mathcal{F}_{KE}$.
2. $\mathcal{P}_j$ sends $(\mathbf{exchange}, sid, \mathcal{P}_i, \mathcal{P}_j, \beta')$ to $\mathcal{F}_{KE}$.
3. Upon receiving both messages, $\mathcal{F}_{KE}$ works as follows.
   1. If $\beta = \beta' = \perp$, $\kappa \leftarrow \{0,1\}^k$.
   2. If $\beta \neq \perp$, $\kappa \leftarrow \beta$.
   3. Else, $\kappa \leftarrow \beta'$.
4. $\mathcal{F}_{KE}$ sends $(\mathbf{Key}, sid, \kappa)$ to $\mathcal{P}_i$ and $\mathcal{P}_j$, and sends $(\mathbf{Key}, sid, \mathcal{P}_i, \mathcal{P}_j)$ to $\mathcal{S}$.

3.2. Construction of π_{TS}

Let $Enc(\cdot,\cdot)$ be an ideal symmetric encryption algorithm and $Dec(\cdot,\cdot)$ be a corresponding decryption algorithm. We denote the concatenation of a and b by $a \cdot b$. We construct π_{TS} by applying hybrid-model as follows (Fig. 5).

Key generation:

1. $\mathcal{TSA}$ sends $(\mathbf{Key}, sid)$ to $\mathcal{F}_{SIG}$ and then executes the key generation process of $\mathcal{F}_{SIG}$.

2. $\mathcal{TSA}$ obtains $(\mathbf{VerificationKey}, sid, \theta)$ and then outputs $(\mathbf{SetUpDone}, sid, \theta)$.

Time-stamp token generation:

1. Time-stamp requester $\mathcal{P}$ sends $(\mathbf{Start}, sid, \mathcal{P}, \mathcal{TSA})$ to $\mathcal{TSA}$.
2. $\mathcal{P}$ and $\mathcal{TSA}$ share session key κ with $\mathcal{F}_{KE}$.
3. $\mathcal{P}$ computes $C = Enc(\kappa, d)$ and then sends it to $\mathcal{TSA}$.
4. Let $M = t \cdot C \cdot ID_{\mathcal{P}}$. $\mathcal{TSA}$ executes the signature generation process of $\mathcal{F}_{SIG}$ with input $(\mathbf{Sign}, sid, M)$ and then obtains $(\mathbf{Signature}, sid, M, \sigma)$.
5. $\mathcal{TSA}$ sends $(\mathbf{Signature}, sid, M, \sigma)$ to $\mathcal{P}$.
6. $\mathcal{P}$ parses M as $M = (t', C', ID'_{\mathcal{P}})$. If $C' = C$ and $ID'_{\mathcal{P}} = ID_{\mathcal{P}}$, $\mathcal{P}$ accepts σ' as the signature. Else, $\mathcal{P}$ rejects it.

Time-stamp token verification:

1. Verifier $\mathcal{V}$ receives $(t, d, ID_{\mathcal{P}}, \sigma, \widetilde{\theta}, \kappa, C)$ as a time-stamp token from a prover.
2. Verifier $\mathcal{V}$ sends $(\mathbf{Verify}, sid, t \cdot C \cdot ID_{\mathcal{P}}, \sigma, \widetilde{\theta})$ to $\mathcal{F}_{SIG}$, and $\mathcal{V}$ executes the verification process of $\mathcal{F}_{SIG}$.
3. $\mathcal{V}$ obtains $(\mathbf{Verified}, sid, t \cdot C \cdot ID_{\mathcal{P}}, f)$ and then $\mathcal{V}$ outputs f if $C = Enc(\kappa, d)$. Otherwise it outputs 0.

In the following, we show that π_{TS} securely realizes $\mathcal{F}_{TS}$ in the UC secure sense.

Theorem 1 *In the $(\mathcal{F}_{KE}, \mathcal{F}_{SIG})$-hybrid model, π_{TS} securely realizes $\mathcal{F}_{TS}$ in the UC secure sense for any adversary.*

Proof: Let $\mathcal{A}$ be an adversary that interacts with entities running π_{TS}. We construct a simulator $\mathcal{S}$ such that the view of any environment $\mathcal{Z}$ of an interaction with $\mathcal{A}$ and π_{TS} is distributed identically to its view of an interaction with $\mathcal{S}$ in the ideal process for $\mathcal{F}_{TS}$. As usual, simulator $\mathcal{S}$ runs an internal copy of $\mathcal{A}$ and of each of the involved parties. All messages from $\mathcal{Z}$ to $\mathcal{A}$ are written to $\mathcal{A}$'s input tape. In addition, $\mathcal{S}$ does the followings.

Simulating key generation: On receiving message $(\mathbf{SetUp}, sid)$ from $\mathcal{F}_{TS}$, $\mathcal{S}$ simulates the key generation protocol of π_{TS}. That is,

1. $\mathcal{S}$ sends $(\mathbf{KeyGen}, sid)$ to $\mathcal{A}$ and then obtains its return $(\mathbf{Verificationkey}, sid, \theta)$.
2. $\mathcal{S}$ records $(\mathcal{TSA}, \theta)$ and sends $(\mathbf{SetUpDone}, sid, \theta)$ to $\mathcal{F}_{TS}$.

Simulating time-stamp token generation: On receiving message $(\mathbf{StampReq}, sid, |d|)$ from $\mathcal{F}_{TS}$ where the requester is $\mathcal{P}$, $\mathcal{S}$ simulates the token generation protocol of π_{TS} before step 4. That is,

1. $\mathcal{S}$ simulates $\mathcal{F}_{KE}$, generates a random session key κ, and sends $(\mathbf{Key}, sid, \mathcal{TSA}, \mathcal{P})$ to $\mathcal{A}$.
2. $\mathcal{S}$ chooses $C \overset{R}{\leftarrow} \{0,1\}^*$ and records the tuple $((\mathbf{StampReq}, sid, |d|), \kappa, C)$ in its list. $\mathcal{S}$ returns $(\mathbf{OK}, sid)$ to $\mathcal{F}_{TS}$.
3. On receiving message $(\mathbf{StampReq}, sid, t)$ from $\mathcal{F}_{TS}$, $\mathcal{S}$ simulates the token generation of π_{TS} after step 3. That is, $\mathcal{S}$ sets $M = t \cdot C \cdot ID_{\mathcal{P}}$ and sends $(\mathbf{Signature}, sid, M)$ to $\mathcal{A}$. On receiving the tuple $(\mathbf{Signature}, sid, M, \sigma)$ from $\mathcal{A}$, $\mathcal{S}$ sends $(\mathbf{StampDone}, sid, M, \sigma, \kappa, C)$ to $\mathcal{F}_{TS}$.

Simulating time-stamp token verification: On receiving message $(\mathbf{Ver}, sid, \alpha)$ from $\mathcal{F}_{TS}$, where $\alpha = (t, d, ID_{\mathcal{P}}, \sigma, \widetilde{\theta}, \widetilde{\kappa}, \widetilde{C})$, $\mathcal{S}$ simulates the verification protocol. That is,

1. $\mathcal{S}$ sends $(\mathbf{Verify}, sid, \widetilde{M}, \sigma, \widetilde{\theta})$ to $\mathcal{A}$ and obtains its return, $(\mathbf{Verified}, sid, \widetilde{M}, \phi)$, where $\widetilde{M} = t \cdot \widetilde{C} \cdot ID_{\mathcal{P}}$.
2. $\mathcal{S}$ simulates $\mathcal{F}_{SIG}$, verifies the signature, and records $(\widetilde{M}, \sigma, \widetilde{\theta}, f)$ in its list. $\mathcal{S}$ returns $(\mathbf{VerDone}, sid, \alpha, \phi)$ to $\mathcal{F}_{TS}$.

Simulating requester corruption: When $\mathcal{A}$ corrupts a requester, $\mathcal{S}$ corrupts that requester in the ideal process, and obtains the set of documents $\{d\}$ that held by the requester. $\mathcal{S}$ sends the documents to $\mathcal{A}$.

Since we assume that $Enc(\cdot,\cdot)$ is an ideal symmetric encryption, the distribution of ciphertext C of (κ, d) in the simulation is identical to the actual one. It is obvious that $\mathcal{S}$ can obtain all secret information of a requester if $\mathcal{A}$ corrupts it; therefore, any $\mathcal{Z}$ can not distinguish which adversary and requesters ($\mathcal{A}$ and $\mathcal{P}$ / $\mathcal{S}$ and $\mathcal{F}_{TS}$) she accesses. This concludes the proof. *Q.E.D*

4. Discussion

In the time-stamp token verification of π_{TS}, verifier $\mathcal{V}$ first verifies signature σ and then checks the validity of ciphertext C. Since we defined that the time-stamp token verification of $\mathcal{F}_{TS}$ follows the verification of $\mathcal{F}_{SIG}$, we need this setting to prove Theorem 1. If verifier $\mathcal{V}$ firstly checks the validity of C, $\mathcal{V}$ can reject the signature without activating the verification of $\mathcal{F}_{SIG}$ in a case of C is invalid; however, the proof fails with this setting. This is because $\mathcal{S}$ can not check the validity of C when $\mathcal{V}$ (i.e., $\mathcal{Z}$) sends a verification query without activating $\mathcal{F}_{SIG}$ (i.e., $\mathcal{A}$).

However, it is natural that $\mathcal{V}$ firstly checks the validity of the ciphertext and then verifies the signature. To implement this setting, we slightly modify the definition of the time-stamp token verification of $\mathcal{F}_{TS}$ as follows.

Time-stamp token verification:
Let $f \in \{0, 1, *\}$, $\phi \in \{0, 1\}$.

1. Verifier $\mathcal{V}$ sends $(\mathbf{Ver}, sid, \alpha)$ to $\mathcal{F}_{TS}$, where $\alpha = (t, d, ID_{\mathcal{P}}, \sigma, \widetilde{\theta}, \widetilde{\rho})$.
2. $\mathcal{F}_{TS}$ executes the following. (1) If $(\widetilde{\theta}, \widetilde{\rho}) = (\theta, \rho)$ and $\mathcal{F}_{TS}$ has already recorded $(\alpha, 1)$, $f \leftarrow 1$. (2) If $(\widetilde{\theta}, \widetilde{\rho}) = (\theta, \rho)$, $\mathcal{S}$ does not corrupt $\mathcal{TSA}$, and $\mathcal{F}_{TS}$ has not recorded $(t, d, ID_{\mathcal{P}}, \sigma', \theta, \rho, 1)$ for $\forall\sigma'$, $f \leftarrow 0$ and $\mathcal{F}_{TS}$ records $(\alpha, 0)$. (3) If $(\widetilde{\theta}, \widetilde{\rho}) \neq (\theta, \rho)$ and $\mathcal{F}_{TS}$ has already recorded $(\alpha, \widetilde{f})$, $f \leftarrow \widetilde{f}$. (4) Otherwise, $f \leftarrow *$.
3. $\mathcal{F}_{TS}$ sends $(\mathbf{Ver}, sid, \alpha, f)$ to $\mathcal{S}$.
4. $\mathcal{S}$ sends $(\mathbf{VerDone}, sid, \alpha, \phi)$ to $\mathcal{F}_{TS}$.
5. If $f = *$, then $\mathcal{F}_{TS}$ sets $f = \phi$ and records (α, ϕ).
6. $\mathcal{F}_{TS}$ sends $(\mathbf{VerDone}, sid, t, d, ID_{\mathcal{P}}, f)$ to $\mathcal{V}$.

With this setting, $\mathcal{S}$ knows the validity of signature σ even if it does not know the corresponding document d. Therefore, $\mathcal{S}$ can simulate the verification.

This modified setting does not provide any advantage to the adversary since she can record all message-signature pairs related to the verification key that she chooses; therefore, she can verify the signatures herself. In most standard digital signature schemes, the verification does not require extraneous communication; therefore, this indeed corresponds to our intuitive notion of a signature verification process. We can apply this to the definition of the verification of $\mathcal{F}_{SIG}$.

In this study, we separate the management of an accurate time t (or a counter value, a hash value, etc.) from the issuing of a time-stamp token, and we defined functionality $\mathcal{F}_{TS}$ for the token issuing. Hence, $\mathcal{F}_{TS}$ does not check the validity of t. If TSA manages both t and token issuing, $\mathcal{F}_{TS}$ is defined such that it checks the validity of t by comparing it with the latest t' recorded in its register. In the linking protocol, a verifier can verify t even if an adversary corrupts the TSA. On the other hand, in the time-stamping protocol based on a digital signature scheme, such as the simple protocol, it is an open question of how to implement the functionality needed for verifying of t.

5. Conclusion

In this paper, we reconsidered the security notions of the time-stamping protocol and defined its functionality based on the UC framework. Our definition follows that of the signature functionality defined by Canetti. In addition, we described the construction of a secure time-stamping protocol, which is similar to the simple protocol, using the key exchange functionality and the signature functionality. We also showed security proof of our proposed protocol in UC framework.

References

[ACPZ01] C. Adams, P. Cain, D. Pinkas and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)," IETF RFC3161.

[BLLV98] A Buldas, P. Laud, H. Lipmaa and J. Villemson, "Time-stamping with Binary Linking Schemes," In Proc. of *CRYPTO98*, LNCS 1462, pp.486-501, Springer-Verlag, 1998.

[BLS00] A. Buldas, H. Lipmaa and B. Schoenmakers, "Optimally Efficient Accountable Time-stamping," In Proc. of *PKC 2000*, LNCS 1751, pp.293-305, Springer-Verlag, 2000.

[C01] R. Canetti, "Universally Composable Security: A New Paradigm for Cryptographic Protocols," available at http://eprint.iacr.org/2001

[C04] R. Canetti, "Universally Composable Signatures, Certification, and Authentication," In Proc. of *the 17th Computer Security Foundations Workshop (CSFW'04)*.

[CF01] R.Canetti and M.Fischlin, "Universally Composable Commitments," Extended version of the paper that appeared at *CRYPTO 2001*.

[CK02] R.Canetti and H.Krawczyk, "Universally Composable Notions of Key Exchange and Secure Channels," Extended version of the paper that appeared at *EUROCRYPT 2002*, pages 337-351.

[HS91] S. Haber and W. S. Stornetta, "How to Time-stamp a Digital Document," *Journal of Cryptology: the Journal of the International Association for Cryptologic Research 3, 2 (1991)*, pages 99-111.

[ISO] ISO/IEC 18014-1, 18014-2 and 18014-3, Information technology – Security techniques – Time-stamping services – Part 1, Part 2, and Part 3.

[Just98] M. Just, "Some Timestamping Protocol Failures", In Proc. of *the Symposium on Network and Distributed Security (NDSS98)*, San Diego, CA, USA, Mar. 1998, Internet Society.

[MO04] S. Matsuo and H. Oguro, “User-side Forward-dating Attack on Time-stamping Protocol,” In Proc. of *the 3rd International Workshop for Applied Public Key Infrastructure (IWAP’04)*, pages 72-83.

[Sure] http://www.surety.com

[UM02] M. Une, and T. Matsumoto, “A Framework to Evaluate Security and Cost of Time Stamping Schemes,” *IEICE Transactions on Fundamentals*, EA85-A, No.1, pp. 125-139, 2002.

Certificate Validation & Revocation

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Augmented CRL Scheme

Lakshminarayanan A
Institute for Infocomm Research, Singapore

Abstract. PKI based applications use digitally signed certificates to bind public keys to user identities. Some digital certificates need to be revoked before their scheduled expiry. Certificate revocation is an important yet burdensome aspect of PKI. In this paper, we present the augmented CRL scheme, a simple yet novel extension to delta-CRLs. Using this scheme, certificate verifying clients need not download base CRLs yet can construct the same using augmented CRLs. We exploit the similarity between X.509 base and delta-CRL data structures. We show that the augmented CRL scheme provides significant bandwidth savings compared to existing CRL based schemes. The amount of downloaded CRL data is also much less compared to earlier schemes. Our scheme is simple, scalable and can easily be integrated into existing CRL based revocation schemes.

Keywords: PKI, revocation, certificate revocation lists (CRLs), delta-CRLs

Introduction

The Internet has become an indispensable communication platform. Security is a key component of Internet based applications. Many network and message security protocols use public key infrastructures (PKI), e.g. SSL, S/MIME, and VPN [4]. PKI use digitally signed public-key certificates. A public-key certificate binds a public key with the identity of the owner of the corresponding private key. X.509 is an industry standard that defines the format of public-key certificates [10], which has been adopted by the IETF PKIX working group [6]. The primary contents of a public-key certificate include a public key, the certificate's unique identifier e.g. serial number, the certificate owner's identity and the certificate expiry date. A certificate can contain extensions, a complete list of standard extensions can be found in [10]. To ensure the authenticated identity of a certificate owner, certificates are digitally signed and issued by a trusted third party called the certificate authority (CA).

The binding between the public key and the owner identity is valid as long as the private key remains under the sole control of its owner. However a private key can be compromised (stolen or lost). If such compromises are not communicated to certificate verifiers, the consequences can be severe. Once a certificate compromise has been detected, the certificate owner is expected to request the revocation authority to revoke the certificate. The revocation authority makes available to certificate verifiers this revocation information. Revocation information should be transmitted reliably and timely to relying clients. Providing timely reliable revocation is expensive. In [1], it is shown that the cost of operating a large-scale PKI will be high primarily due to revocation services. Well known revocation schemes include periodically issued certificate revocation lists (CRLs) [4], online certificate status protocol (OCSP) [9], certificate revocation system [8], certificate revocation trees [7], trusted directories, broadcast CRLs [4] and short lived certificates [5].

Revocation schemes based either on periodically issued CRLs or OCSP are by far the most popular. In this paper, we restrict our focus to CRL based schemes. An important problem of CRL schemes is scalability. As PKI population size increases, CRL sizes also increase and consequently bandwidth requirements imposed on CRL distribution servers. Segmented CRLs, delta-CRLs or over-issued CRLs can be used to alleviate this problem. These schemes have some disadvantages. In this paper we present the augmented CRL scheme which is superior to existing CRL based schemes, yet retains the simplicity of traditional CRLs. We exploit the similarity of data structures between traditional X.509 base and delta-CRLs. Augmented CRLs have small sizes even for large PKI populations and there is no need to download base CRLs. Augmented CRLs use X.509 CRL extensions and hence can be easily integrated into existing CRL based revocation schemes (in this paper, certificate and CRL refers to X.509 version 3 certificate and X.509 version 2 CRL respectively).

1. Certificate Revocation Lists (CRL)

A certificate revocation list is a time-stamped digitally signed list of serial numbers (or other certificate identifiers) of unexpired certificates that have been revoked by a revocation authority (usually the CA which issued the certificate). This digitally signed list is made available on CRL distribution servers. CRLs are updated and issued at regular intervals even if the revocation list has not changed. Distribution schemes that deliver CRLs to clients can be built based on a variety of protocols such as LDAP, HTTP and FTP [10]. Revoked certificates remain in the CRL list until they expire. CRLs contain two types of extensions - CRL entry extensions that are associated with each revoked certificate e.g. revocation reason code and CRL general extensions which apply to the entire CRL e.g. CRLNumber. The complete list of standard CRL general and entry extensions can be found in [10]. The key elements of the X.509 v2 CRL are shown in Figure 1.

CRL Header (~50 bytes)
- Issuer's name: 32 bytes (if X.500 name used)
- Date/time of CRL issuance (thisUpdate) : 6 bytes
- Date/time of next CRL issuance (nextUpdate): 6 bytes

List of revoked certificates (9 bytes per revoked certificate)
- Serial number : 3 bytes
- Revocation date : 6 bytes
- CRL entry extensions (e.g. revocation reason)

CRL general extensions (e.g. CRL Number)
Signature of CRL issuer: (~130 bytes - 1024 bit RSA)

Figure 1. Key elements of X.509 v2 CRL

1.1. PKI Model

To analyze CRL performance, we adopt Cooper's PKI model and assumptions [2]. They are

- PKI population is large and hence client certificate validation requests arrive independent of each other
- Clients cache CRLs and don't download a new CRL until the cached CRL's nextUpdate
- There is one CRL distribution server (If there are *K* such servers, the bandwidth required per server is reduced by a factor of *K*)
- Once a certificate is revoked, its status is reflected in the CRL until the certificate expires. A revoked certificate will remain in a CRL for approximately half its lifetime [2]
- Base CRL reach steady state size
- N = 300,000 (total users in PKI population)
- P = 365 days (certificate validity period)
- $v = 10$ certificates validated per day by each client
- $R_{revoke} = 1000$ revocations per day
- $R_{expiry} = 1000$ revoked certificates expire per day
- $R_{revoke} = R_{expiry}$ after base CRLs attain steady state size. This probably won't hold true in real-world settings but since base CRLs have steady state size, analysis is simpler. Cooper [2, 3] implicitly makes the same assumption

If client validation requests arrive independent of each other, an exponential inter-arrival probability density function can be used to model such arrivals. Using this assumption, Cooper [2] derived the request rate (CRL_{Req}) for CRLs at time *t* (CRLs downloaded from a CRL distribution server) given by Eq. (1).

$$CRL_{Req}(t) = Nve^{-vt} \tag{1}$$

Given the assumption that revoked certificates remains in a CRL for half their lifetime, the approximate size of a complete CRL ($S_{complete}$) is given by Eq. (2). This includes issuer's name, date and time fields, certificate serial numbers, signature bytes and ASN.1 packaging data.

$$S_{complete} = 180 + 4.5PR_{revoke} \tag{2}$$

1.2. Existing CRL Schemes

A ***complete CRL*** contains the complete list of all unexpired revoked certificates of a PKI population**.** A trivial revocation scheme would require the revocation authority (henceforth referred as the CRL issuer) to publish complete CRLs in its distribution servers and clients to download these complete CRLs. If the PKI population is large, CRLs become large. Downloading large CRLs imposes high communication costs on the CRL issuer. If clients have limited bandwidth capability e.g. dial-up connections, long download times can be annoying. The factors influencing complete CRL size include PKI population size, revocation rate and certificate validity period, the former two usually beyond the control of the CRL issuer. We analyze and compare popular CRL based revocation schemes based on the following performance metrics.

- CRL size
- Peak CRL request rate
- Peak CRL bandwidth
- Total amount of CRL data downloaded

Short-lived certificates: Since an expired certificate can be removed from a CRL, it is tempting to set shorter validity periods for certificates to reduce the size of complete CRLs or even use very short-lived certificates [5] wherein no revocation is needed. This is however not preferable because frequent certificate generation and reliable updating of certificates leads to higher costs. Users will also be inconvenienced by frequent certificate renewals. If short-lived certificates are used for non-repudiation purposes, this will lead to higher archival costs.

Segmented CRLs: It is possible to arbitrarily divide the certificate population into many partitions, each partition being associated with a CRL distribution point [10]. Verifying clients obtain the CRL distribution server's location from this CRL distribution point extension. While segmenting CRLs does not reduce peak CRL request rate, it reduces the size of each CRL downloaded reducing the peak bandwidth [2]. However, if relying clients' operate off-line, segmentation is not useful. Relying parties that operate off-line will not know which certificates they will be validating at the time they obtain the CRLs. They need to download all the segmented CRLs [2]. Even if clients are on-line - in the worst-case, clients need to download CRLs from all distribution points.

Over-Issued CRLs: Traditionally, a new CRL is not issued until the time specified in the nextUpdate field of the currently issued CRL has been reached. Consequently, a relying party with an unexpired CRL in its cache will not request a new CRL from the repository until the nextUpdate time. This leads to high peak request rates just after a new CRL is issued. However, if CRLs are over-issued, i.e. a new CRL is issued before the previous one expires (before the nextUpdate time of the previous CRL has been reached), some relying clients will retrieve this new CRL while others continue to use previously issued yet valid CRL. If each issued CRL is valid for the same length of time, then relying clients that retrieved the new CRL will still have valid CRLs in their caches when the original CRL expires [2]. Over-issued CRLs partition the CRL issuance interval into time segments. Cooper's analysis [2] shows that over-issued certificates are effective in reducing peak CRL request rate. However the total amount of data downloaded from CRL distribution servers remains the same.

Delta-CRLs: A delta-CRL (CRL_{Δ}) is a time-stamped digitally signed revocation list of changes that occurred since the issuance of a prior base CRL. A base CRL is complete CRL that contains a complete list of revoked certificates, to which the revocation list in the delta-CRL needs to be applied to produce the latest list of revoked certificates. X.509 base and delta-CRLs have similar data structures. A delta-CRL uses its DeltaCRLIndicator extension [10] to identify the base CRL it refers to, the DeltaCRLIndicator containing the base CRL's CRLNumber. Clients download delta-CRLs frequently and occasionally a base CRL (Figure 2). Clients then construct an up-to-date list of unexpired revoked certificates. The request rate for delta-CRLs is given by Eq. (1). The size of a delta-CRL depends on the time it is issued. If issued just after issuance of base CRL, it is small and if issued just before issuance of base CRL, it is large. If *T* is the base CRL issuance interval and *M* delta-CRLs are issued between two base CRLs (at regular intervals), the approximate size of the j^{th} delta-CRL ($S_\Delta(j)$) is given by Eq. (3). If the base CRL issuance interval is long, base CRLs need not be downloaded frequently. Then the size of delta-CRLs becomes larger and consequently the bandwidth required for downloading delta-CRLs increases. Optimum base and delta-CRL issuance intervals should be chosen depending on the PKI application.

If a client possesses a secure trusted repository, it is possible to completely eliminate the need to download base CRLs. The client keeps updating the list of revoked certificates in its trusted secure repository using information obtained from delta-CRLs. However, this is not practical. Trusted secure repositories in environments such as desktops, mobile clients where such trusted repositories need to be maintained for long periods of time will be expensive. Even if such repositories are available, they cannot be shared by other clients without proper trust relationships. In this paper, we assume that clients do not possess trusted secure repositories.

$$S_\Delta(j) = 180 + 9(jT/M)R_{revoke}, j = 1 \text{ to } M \quad (3)$$

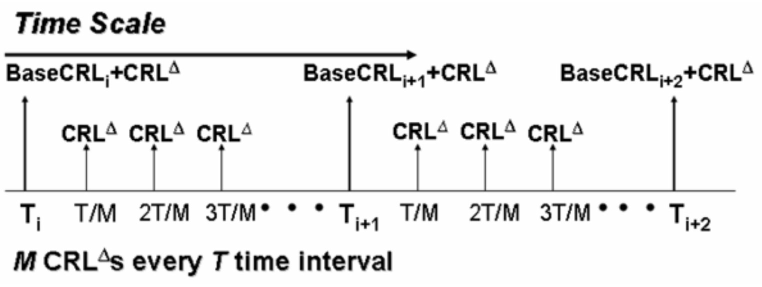

Time Scale
BaseCRLi+CRLΔ
BaseCRLi+1+CRLΔ
BaseCRLi+2+CRLΔ
CRLΔ CRLΔ CRLΔ
CRLΔ CRLΔ CRLΔ
Ti T/M 2T/M 3T/M • • • Ti+1 T/M 2T/M 3T/M • • • Ti+2
M CRLΔs every T time interval

Figure 2. Delta-CRL issuance time-line

Sliding window Delta-CRLs: Cooper [3] showed that if delta-CRLs are issued in the traditional manner, i.e. base CRLs issued at regular intervals and delta-CRLs issued regularly but more frequently, the performance gain of using delta-CRLs is not significant. This is because, at the time of issuance of the first delta-CRL (after base CRL issuance), the base CRL request rate is not small. Cooper's sliding window delta-CRL scheme [3] uses a large window size (time interval between delta-CRL and the referenced base CRL issuance) for all issued delta-CRLs. This reduces the peak base CRL request rate considerably. Since a large window size is used, the size of delta-CRLs becomes large making delta-CRL contribution to peak bandwidth significant. Over-all the sliding window scheme provides significant bandwidth savings. Cooper derived formulas for peak bandwidth and optimum window sizes using his PKI model [2]. Clients using the sliding window schemes use local repositories for updating the latest revocation list. These local repositories not only should be secure but cannot be shared with other relying clients. Table 1 compares the performance of different CRL schemes explained in this section.

Table 1. Various CRL schemes' performance

CRL Scheme	**CRL issuance interval (hrs)**	**CRL size (KB)**	**Peak request rate (/sec)**	**Peak bandwidth (KB/sec)**	**Total data transferred (GB/day)**
Base CRL only	BaseCRL: 12	1604.2	34.7	**55665.7**	917.9
Segmented CRL (Ten segments)	BaseCRL: 12	160.4	34.7	**5566.6**	917.9(Worst case) 91.8 (Best case)
Over-issued CRL	Base CRL: 12 (over-issued every 4 hours)	1604.2	11.1	**17806.6**	917.9
Traditional delta-CRL	BaseCRL: 12 DeltaCRL: 4	BaseCRL: 1604.2 DeltaCRL: 1.6 to 4.6	BaseCRL : 6.6 DeltaCRL: 34.7	BaseCRL: 10587.7 DeltaCRL: 56.9 Combined: **10644.6**	> 917.9
Sliding window	BaseCRL: 12 DeltaCRL: 4 (window size 20 hours)	BaseCRL: 1604.2 DeltaCRL: 7.5	BaseCRL: 0.008 DeltaCRL: 34.7	BaseCRL: 12.8 DeltaCRL: 260.3 Combined: **273.1**	> 459.0
Sliding window	BaseCRL: 12 DeltaCRL: 1 (window size 18 hours)	BaseCRL: 1604.2 DeltaCRL: 6.8	BaseCRL: 0.019 DeltaCRL: 34.7	BaseCRL: 30.8 DeltaCRL: 236.0 Combined: **266.8**	> 459.0
Sliding window with over-issued delta-CRLs	BaseCRL: 12 DeltaCRL: 4 (DeltaCRLs over-issued every hour. Window size 21 hours)	BaseCRL: 1604.2 DeltaCRL: 7.9	BaseCRL: 0.009 DeltaCRL: 17.2	BaseCRL: 14.4 DeltaCRL: 135.9 Combined: **150.3**	> 459.0

1. Cooper's PKI model [2]. Rounded-off values obtained using appropriate formulae from [2, 3]
2. Optimal window sizes selected for the sliding window schemes using formulae from [3]
3. In sliding window schemes [3], at least one BaseCRL downloaded every day by each client

2. Augmented CRL Scheme

X.509 base CRLs and delta-CRLs have similar data structures. We use this similarity to construct augmented CRLs. The attributes of a periodically issued X.509 CRL that change with every fresh CRL issuance are

- Date and time of CRL issuance (thisUpdate)
- Date and time of next CRL issuance (nextUpdate)
- CRL number
- Latest list of revoked certificates
- Signature over CRL contents

Changes in the latest list of revoked certificates include new revocations reported between issuance of previous CRL and the freshly issued CRL as well as deletions of expired certificates. This list can be an ordered list (e.g. certificate serial numbers in ascending order) indicated using the ordered list CRL extension [10]. To construct augmented CRLs, we make the following assumptions.

- List of revoked certificates is ordered
- All CRLs have CRLNumbers. Delta-CRLs refer to base CRLs using the base CRL's CRLNumber
- When a delta-CRL is issued, a complete CRL is also issued at the same time
- CRLNumber of the complete CRL and delta-CRL issued at the same time is same
- thisUpdate and nextUpdate fields of the complete CRL and delta-CRL issued at the same time are same
- The CRL type (complete, delta) is determined by the respective CRL extension e.g. delta-CRLs have DeltaCRLIndicator extension
- CRL extensions do not change with time
- CRLs expire when the CRL issuer's signing key expires

Using the latest issued delta-CRL (and the base CRL that this delta-CRL refers to), a client possesses (or can construct) all attributes of the complete CRL (issued at the same time as the delta-CRL) except for the digital signature of the CRL issuer over the contents of the complete CRL. ***An augmented CRL (ACRL) is a delta-CRL which contains the CRL issuer's digital signature over the complete CRL (issued at the same time as the augmented CRL) as an additional CRL general extension*** (this CRL general extension will henceforth be called the signature bytes extension).

Since the data structures of X.509 base CRLs and augmented CRLs are very similar, a client with an augmented CRL (and the base CRL this augmented CRL refers to) possesses all the attributes of the complete CRL (issued at the same time as the augmented CRL) such as CRLNumber, thisUpdate and nextUpdate. The list of revoked certificates contained in the complete CRL is an ordered list. It is the union of the revocation list of the base CRL that the augmented CRL refers to and the list contained in the augmented CRL (we ignore deletions from this list due to certificate expiry. This aspect is addressed soon). The digital signature of the CRL issuer over the complete CRL is present in the signature bytes extension. So a client just needs to regularly download augmented CRLs to locally construct complete CRLs issued at the same time.

An augmented CRL is many times smaller than a complete CRL and hence will result in considerable bandwidth savings. An augmented CRL will be slightly larger than a delta-CRL because of the signature bytes extension (~ 130 bytes for 024 bit RSA signature). Locally constructed complete CRLs can be shared with other clients since they are identical to complete CRLs.

2.1. Handling Expired Certificates

A CRL contains serial numbers of unexpired revoked certificates. Once a revoked certificate has expired, it need not be present in the CRL. In existing CRL schemes, the CRL issuer regularly removes expired certificates from base CRLs. But a client which downloads only augmented CRLs has no information about expired revoked certificates. To address this issue, we use two types of augmented CRLs - ordinary augmented CRLs and augmented milestone CRLs. All augmented CRLs carry the signature bytes CRL general extension. ***An augmented milestone CRL carries an additional X.509 CRL general extension containing the list of the revoked certificates that have expired since the issuance of the previous augmented milestone CRL.*** This extension also indicates when the next milestone CRL will be issued. An augmented CRL without this additional CRL extension is an ordinary augmented CRL (henceforth an ordinary augmented CRL will be referred to as augmented CRL and an augmented milestone CRL as milestone CRL).

A milestone CRL is issued every T interval and an augmented CRL every T/M interval (M augmented CRLs every T interval). A complete CRL is issued every time an augmented CRL (both ordinary and milestone) is issued and complete CRLs issued with milestone CRLs serve as reference base CRLs. An augmented CRL refers to a milestone CRL in the same way a traditional delta-CRL refers to a base CRL - using its DeltaCRLIndicator extension. The approximate size of the j^{th} ordinary augmented CRL is given by Eq. (4) and that of a milestone augmented CRL by Eq. (5). The size of a milestone CRL is marginally larger since it carries the list of expired revoked certificates (3 bytes for each expired certificate's serial number). If a client possesses the i^{th} Base CRL, construction of a complete CRL ($CompleteCRL_{i+1,\,j}$ j=1 to M) issued along with the j^{th} ordinary augmented CRL at time $T_{i+1+j*T/M}$ is shown in Figure 3.

In Figure 4 assume a client has performed its last certificate validation sometime after T_i *(so* client possesses $BaseCRL_i$) and its next CRL request is just before T_{i+2}. If the client uses the traditional delta-CRL scheme, it needs to download $BaseCRL_{i+1}$ at T_{i+1} as well as the latest delta-CRL. However, with the augmented CRL scheme, the much smaller milestone CRL issued at T_{i+1} is sufficient to create $BaseCRL_{i+1}$. Hence the client needs only two augmented CRLs to construct the latest issued complete CRL, one milestone CRL issued at T_{i+1} ($MilestoneCRL_{i+1}$) and the latest augmented CRL ($Augmented\ CRL_{i+1,j}$). If the client has $BaseCRL_{i-1}$ instead of $BaseCRL_i$, the client obtains two milestone CRLs ($MilestoneCRL_i$ and $MilestoneCRL_{i+1}$) and one augmented CRL ($Augmented\ CRL_{i+1,\ j}$). The client first constructs $BaseCRL_i$, then constructs $BaseCRL_{i+1}$ and finally the latest complete CRL ($CompleteCRL_{i+1,\,j}$). Hence a client can iteratively construct complete CRLs without downloading any base CRL.

$$S_{Augmented}(j) = 310 + 9(jT/M)R_{revoke},\ j = 1 \text{ to } M \tag{4}$$

$$S_{Milestone} = 310 + (9R_{revoke} + 3R_{expiry})T \tag{5}$$

Step 1: List of revoked certificates of BaseCRL$_{i+1}$ =
Revocation list of BaseCRL$_i$
+ Revocation list of MilestoneCRL$_{i+1}$
− List of expired certificates in MilestoneCRL$_{i+1}$
Step 2: Construct baseCRL$_{i+1}$
Step 3: List of revoked certificates of CompleteCRL$_{i+1,\,j}$ =
Revocation list of BaseCRL$_{i+1}$
+ Revocation list of Augmented CRL$_{i+1,\,j}$
Step 4: Construct CompleteCRL$_{i+1,\,j}$

Figure 3. Constructing a complete CRL

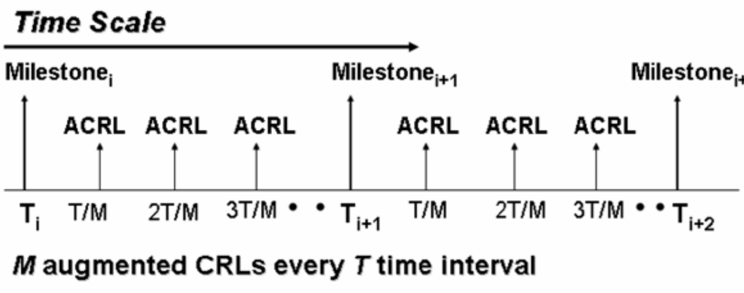


Figure 4. Augmented CRL issuance time-line

Table 2 shows the performance of the augmented CRL scheme as applied to Cooper's PKI model [2]. Peak milestone CRL bandwidth occurs every time a fresh milestone CRL is issued. Peak augmented CRL bandwidth occurs when the first augmented CRL is issued, i.e. the augmented CRL issued *T/M* time interval after issuance of a milestone CRL. Peak augmented CRL bandwidth includes bandwidth required to download the latest milestone CRL (milestone CRL bandwidth at this time is reduced to $Nve^{-vT/M}$). A comparison of Table 1 and 2 shows that our augmented CRL scheme performs much better than earlier CRL schemes including Cooper's sliding window CRL scheme [3]. The total amount of CRL data downloaded is at least an order of magnitude lesser. Since augmented CRL sizes are small, clients with limited bandwidth capability can still easily download these CRL files.

In some applications, certificate validation requests do not arrive independent of each other, e.g. email applications where messages are frequently and repeatedly exchanged. Email applications also experience high transient loads, especially early morning office hours. In the sliding window scheme, clients need to download base CRLs albeit at very low request rates. But if the assumptions in [2, 3] don't hold true, base CRL peak bandwidth is likely to be higher than the values obtained using Cooper's analysis. Clients using augmented CRLs do not download base CRLs, hence will be able to better handle situations where assumptions made in [2,3] don't hold or at high transient loads. If augmented CRLs are over-issued, the peak bandwidth will reduce further. Since our augmented CRL scheme's peak bandwidth (without over-issuing) is much lesser than the sliding window CRL scheme (with over-issuing), we do not discuss over-issued augmented CRLs anymore in this paper.

Table 2: Augmented CRL scheme performance

CRL Issuance interval (hrs)	**CRL Size (KB)**	**Peak request rate (/sec)**	**Peak bandwidth (KB/sec)**	**Total data downloaded (GB/day)**
Milestone: 12 Augmented: 4	Milestone: 6.2 Augmented: 1.8, 3.2	Milestone: 34.7 Augmented: 34.7 (+ Milestone: 6.6)	Milestone: 215.1 Augmented: 103.4	10.9
Milestone: 8 Augmented: 4	Milestone: 4.2 Augmented: 1.8	Milestone: 34.7 Augmented: 34.7 (+ Milestone: 6.6)	Milestone: 145.7 Augmented: 90.2	7.2
Milestone: 4 (All CRLs are milestone CRLs)	Milestone: 2.3	Milestone: 34.7 (+ previous milestone: 6.2)	Milestone: 94.1	3.9
Milestone: 4 Augmented: 2	Milestone: 2.3 Augmented: 1.0	Milestone: 34.7 Augmented: 34.7 (+Milestone: 15.1)	Milestone: 79.8 Augmented: 69.4	5.1
Milestone: 4 Augmented: 1	Milestone: 2.3 Augmented: 0.7, 1.0, 1.4	Milestone: 34.7 Augmented: 34.7 (+Milestone: 22.9)	**Milestone: 79.8** **Augmented: 77.0**	5.6

1. Cooper's PKI model [2]
2. Peak bandwidth for augmented CRLs includes the bandwidth for the latest milestone CRL
3. To calculate the total amount of downloaded CRL data, we assume the worst-case scenario: largest augmented CRLs are downloaded in addition to all milestone CRLs

2.2. Missed Milestone CRLs

Clients can become inactive and miss downloading milestone CRLs. In the traditional delta-CRL scheme, a client needs to download just one base CRL and a delta-CRL to construct the latest list of revoked certificates regardless of how long it has been inactive. But with our augmented CRL scheme, a client needs to download all missed milestone CRLs. Without all milestone CRLs, the client cannot construct the latest base CRL. The performance penalty of downloading missed milestone CRLs is not severe. Unless the client has been inactive for too long, the total size of milestone CRLs to be downloaded remains small. If we use Cooper's PKI model [2], the size of a base CRL is 1604.2 KB. Assuming that a milestone CRL is issued every 4 hours, its size will be 2.3 KB. A client needs to be inactive for almost 116 days until the cost of downloading all previously issued milestone CRLs becomes higher than downloading a base CRL. The CRL distribution server stores all milestone CRLs that are issued by the CRL issuer. If milestone CRLs are issued every 4 hours, the CRL distribution server stores 4.9 MB (6*365*2.3 KB) of milestone CRLs every year This is a small price to pay (hard-disk cost is cheap) for the considerable bandwidth savings. We use a simple protocol between client and CRL distribution server for obtaining the latest CRL information.

- Client sends the CRLNumber of the last base CRL it retrieved (or possesses)
- The server sends back all milestone CRLs and the latest augmented CRL that are necessary for client to compute the latest complete CRL
- The client iteratively constructs base CRLs until it constructs the latest complete CRL
- If client sends -1 (instead of a CRLNumber), the latest base CRL and the latest complete CRL are sent back by the server

After a client has received all necessary milestone CRLs and the latest augmented CRL, it verifies the signature over each CRL. Each milestone CRL contains the date and time the next milestone CRL is issued, using which a client verifies whether it has obtained all necessary milestone CRLs. The client then iteratively constructs all previous base CRLs and the latest complete CRL.

2.3. Integration with Existing X.509 CRL Schemes

X.509 version 2 CRL extensions can be designated critical or non-critical. If a relying client doesn't recognize a non-critical extension, the client ignores the extension but can still use other attributes of the certificate. However, if an extension is marked critical, then a relying client must be able to recognize the extension if it needs to use any part of the certificate [10]. An augmented CRL carries two CRL extensions.

- The signature bytes CRL general extension containing the signature bytes of CRL issuer (signature over complete CRL issued at the same time as augmented CRL)
- The expired certificate list CRL general extension - present only in milestone CRLs, containing the list of expired revoked certificates (expired since the previous milestone CRL)

For backward compatibility, all augmented CRL extensions should be designated non-critical. If a certificate-verifying client doesn't recognize these non-critical extensions, it can continue using the traditional delta-CRL scheme. On the other hand, a client capable of recognizing these extensions benefits from the augmented CRL scheme. Since augmented CRLs use X.509 CRL extensions, our scheme can be integrated into existing CRL schemes without major modifications.

3. Conclusion

In this paper, we have presented a much simpler yet effective revocation scheme based on augmented CRLs. This scheme performs better than earlier CRL schemes including Cooper's sliding window scheme [3]. We are currently improving our scheme and analysis to build an online certificate status scheme using augmented CRLs. To summarize, the key advantages

- Augmented CRLs are small, even for large PKI populations providing significant bandwidth savings. Consequently, CRL download times are short benefiting bandwidth constrained devices
- Peak bandwidth and total CRL data downloaded are much lower
- Clients do not need secure trusted repositories to construct up-to-date list of revoked certificates. Clients can locally construct base CRLs without having to download them
- Since locally constructed base CRLs' integrity is protected by the CRL issuer's signature, augmented CRLs can be shared with other clients
- There is no need to alter the CRL format or use non-CRL revocation schemes e.g. [7, 8]. Using non-critical X.509 extensions, augmented CRLs can easily be integrated into existing X.509 CRL schemes

References

[1] S. Berkovits, S.Chokhani, J.A. Furlong, J.A. Geiter, J.C. Guild, "Public key infrastructure study: Final report", produced by the MITRE Corporation for NIST, April 1994.

[2] D. A. Cooper, "A model of certificate revocation", Proceedings of the 15th annual computer security applications conference, 1999, http://csrc.nist.gov/pki/PKImodels/

[3] D. A. Cooper, "A more efficient use of delta-CRLs", Proceedings of the IEEE symposium on security and privacy, May 2000, http://csrc.nist.gov/pki/PKImodels/

[4] W. Ford and Micheal S. Baum, "Secure electronic commerce", second edition, 2001, Prentice Hall.

[5] Y.K. Hsu, S.P. Seymour, "An internet security framework based on short-lived certificates", IEEE internet computing, March/April 1998.

[6] R. Housley, W. Ford, W. Polk, and D. Solo, "Internet X.509 public key infrastructure certificate and CRL profile". RFC 2459, January 1999

[7] P.C. Kocher, "On certificate revocation and validation", Proceedings of the international conference on financial cryptography, Volume 1465, LNCS, Springer-Verlag, 1998.

[8] S. Micali, "Enhanced certificate revocation system", Technical memo, MIT/LCS/TM-542b, March 1996.

[9] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP", RFC 2560, June 1999.

[10] ITU-T. "Information technology - Open systems interconnection - The directory: Public-key and attribute certificate frameworks", ITU-T Recommendation X.509 (V4), 2000.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Speeding up X.509 Certificate Path Validation

Shingo Hane, Takahiro Fujishiro, Yoko Hashimoto, Tadashi Kaji, Kondo Katsuhiko and Satoru Tezuka
Hitachi, Ltd., 890, Kashimada, Saiwai-ku, Kawasaki-shi, Kanagawa-ken, 212-8567 Japan
E-mail: {hane, fujisi, kumagai, t-kaji, ka-kondo, tezuka}@sdl.hitachi.co.jp.

Abstract. A fast algorithm to execute path validation of the X.509 Certificate was developed. The algorithm used not only certificates and certificate revocation information but also certification path information as cached information in order to speed up transactions. The effect of it was confirmed by the experiment of the server system that had the fast algorithm of certification path validation under the private test environment.

Keywords. PKI, X.509 Certificate, Certification Path, Path Validation, Security

Introduction

One of the applications of PKI (Public Key Infrastructure) represented by ITU-T X.509 [1] is used in authentication of servers and exchanging cryptographic secret keys. It is used as SSL or TLS protocols [2,3]. Due to the development of legal systems so-called Electric Signature Law in many countries, PKI is also used as electric signature for concluding contracts and for authentication of users[4,5]. And PKI is applied in many other cases such as using for IEEE 802.1X network authentication and device authentication etc. Because of the increase of PKI application and utilized opportunities, a lighter loading and higher speed PKI operation method is demanded by the application users.

Encryption processing as generating and verifying a signature value is a necessary operation to use PKI. And one of the prevailing methods for fast processing is considered as improvement of algorithm of encryption processing. The network access processing is also sometimes needed to get revocation information when checking validity of a certificate. Improvement of data cache and searching algorithm are effective to speed up this process. We hereunder describe about development of fast algorithm to validate a certificate by storing and reusing network cache and intermediate processing state.

1. Verifying Certificate

When authenticating by using PKI, it is necessary to verify the signature value and the certificate. To verify a value of a PKI digital signature, the public key recorded in the certificate is used for verification. To verify a certificate, it is necessary to confirm whether the certificate is issued correctly by a trusted Certification Authority (CA). And a certificate would be revoked in case of a change in the certificate record or the private key compromise. Therefore a status of a certificate validity needs to be confirmed even it was issued correctly. If only verification of a digital signature value is executed and validation of the corresponding certificate isn't executed, it is impossible to detect an incorrect or revoked certificate and thus assurance of authenticity by PKI cannot be realized. Considering the above, it is important to verify certificates as well as to verify digital signatures when authenticating by using PKI. There are two models in validation approach of certificates, the multiple trust point distribution model and the cross certificate model. First, we describe about the multiple trust point distribution model and the cross certification model, and next, describe the method of certificate validation under the cross certification model and the method of processing it under the client server system.

1.1. Multiple Trust Point Distribution Model

When processing a certificate validation, several CA certificates are stored in the validation system as Trust Anchor (TA) certificates under the multiple trust point distribution model. And the CA certificates for validation are switched corresponding to the subscriber certificate. This model is called Multi Trust Point Distribution Model because multi CAs are trusted by users and their certificates are distributed and stored on validation system to be used for validation. It is the model used by ordinary web browsers to validate SSL server certificate.

1.2. Cross Certification model

Under cross certification model, one or few CA certificate is distributed and stored as a TA certificate in the validation system to be used for certificate validation. The relationship between CAs called cross certification is used to validate a subscriber certificate. It is the model adopted by public section such as Governmental Public Key Infrastructure [6,7] and the Public Certification Service for Individuals in Japan [8].

1.3. Certificate Validation Method

A certification chain made of certificates is called a certification path. And a certification path is verified as a certificate validation under the cross certification model. Figure 1 shows an example of building a certification path to validate a End Entity (EE) certificate as a subscriber certificate. The certificate validation function builds a certification path from TA certificate to EE certificate and validates it. In this example, there are three cross certificates and four Certificate Revocation List (CRL)/Authority Revocation Lists (ARLs) to be used as certification information to validate the EE certificate. Each certificates and CRL/ARLs are searched and collected from the directory server called repository.

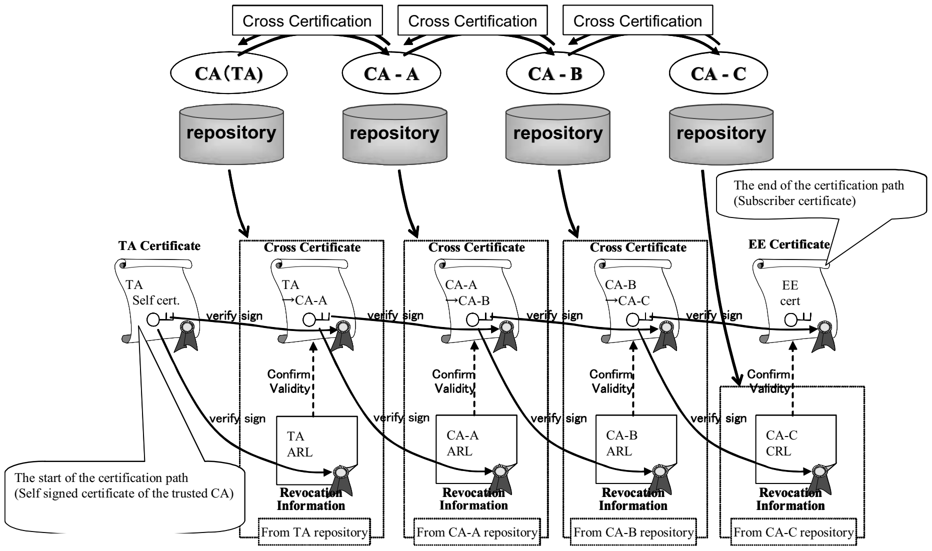


Figure 1. An example of a validation of a certification path

1.4. Client and Server system

As a Certificate validation under the cross certification model needs many processes, it is heavy loading for many client systems to validate all certificates in a certification path every time they receive it. So it is considered effective to operate a certification validation processing at a server to reduce loading of client, and thus the protocols between a client and a server and the validation method are investigated [9,10]. As the protocols for certificate validation, there is one using the Online Certificate Status Protocol (OCSP) [11] extension fields produced by Japanese Government [7], and another one named Simple Certificate Validation Protocol (SCVP) [12] discussed in IETF.

We herein call a server which operates certificate validation processing as CVS (Certificate Validation Server) and made the research on it. When processing on a server, validation requests of the same certificate are repeatedly sent to a server by several clients. Therefore, storing and administrating necessary data such as certificates and CRL/ARLs through this validation processing is expected to speed-up the validation of certificates. We hereunder report our development of several speed-up methods by using the above cache algorithms, and the measurement of its effect.

2. Speeding up of certificate validation

The following three methods were examined this time to speed-up certificate validation function of CVS.

2.1. Speeding up by certificate cache

The certificate validation function is speeded up by caching certificates and revocation information that the function has obtained and used once. In the past model, certification validation function obtained the necessary certificates and revocation information by accessing to the directory server each time of validation. The improved certificate validation algorithm is speeded up by reducing the access to the directory server by caching the CA certificates or revocation information as shown in figure 1.

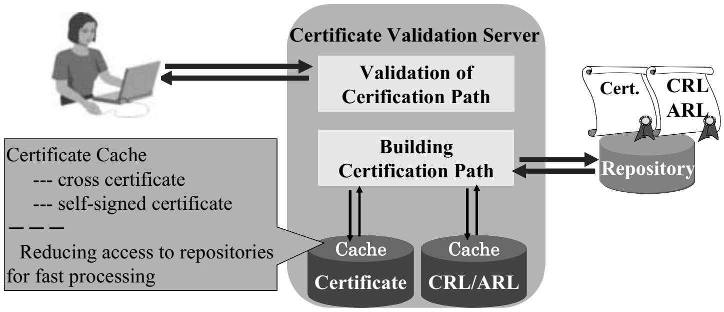

Certificate Validation Server
Validation of Cerification Path
Building Certification Path
Cert.
CRL ARL
Repository
Certificate Cache
--- cross certificate
--- self-signed certificate
Reducing access to repositories for fast processing
Cache Certificate
Cache CRL/ARL

Figure 2. Speeding up by certificate cache

2.2. Speeding up by certification path cache

The certificate validation function is speeded up by caching the certification path that was once built. The certification path is built based on the record of certificates, but there is no optimized method for it. Under the current system, it is necessary to collect all the certificates as possible to build the certification path. So the building processing of the certification path is expected to be a heavy burden. Thus the improved certification validation algorithm in figure 3 caches the certification path which was once built. By caching certification paths in addition to certificates and CRL/ARLs as described in 2.1, the building process of the certification path executed each time of requests can be simplified and the certificate validation function can be speeded up.

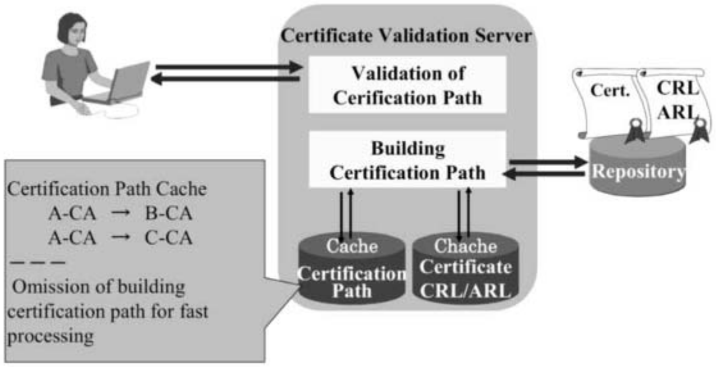

Certificate Validation Server
Validation of Cerification Path
Building Certification Path
Cert.
CRL ARL
Repository
Certification Path Cache
A-CA → B-CA
A-CA → C-CA
Omission of building certification path for fast processing
Cache Certification Path
Chache Certificate CRL/ARL

Figure 3. Speeding up by certification path cache

2.3. Speeding up by building revocation table

The certificate validation function is speeded up by building a revocation table from CRL/ARLs as shown in figure 4. In order to confirm the revocation status of a certificate, it is necessary to confirm if the serial number of the certificate is recorded in the CRL/ARL or not. As shown in left part of figure 4, the serial numbers of revoked certificates are generally recorded in order of revocation date in CRL/ARL as revocation information. So it is necessary to make sure that the serial number is not listed in the revocation list by checking all the serial numbers from the first to the last record in the list, in order to confirm the revocation status of certificate not listed in the revocation information. Such procedure has little impact on processing time as long as the contents in the list are limited. But it would have significant impact on processing time in case of issuing revocation information for more than 10 thousand cases such as the Public Certification Service for Individuals in Japan.

In the certificate validation server, the registered record in the revocation list is sorted by serial number as shown in right part of figure 4, and a serial number is searched by bisection method. The bisection method needs less processing compared to searching all records. The difference of the processing time is remarkable when the record is huge. The speed-up of certificate validation function is realized by sorting the revocation list and building the revocation table.

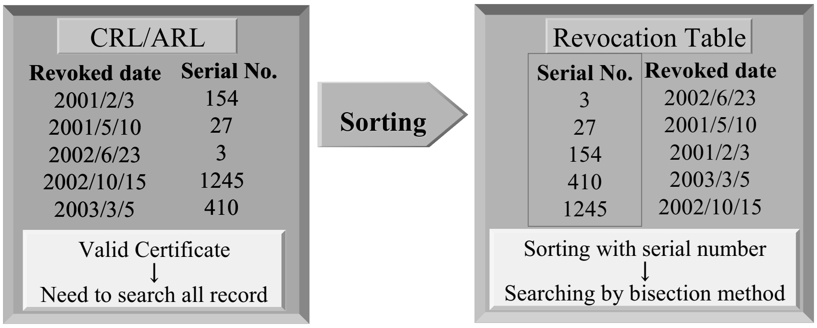


Figure 4. Speeding up by building revocation table

3. Estimate method of speeding up

3.1. Estimate method

To estimate speed-up method of certificate validation processing described in section 3, the certificate validation algorithm was integrated into a CVS and its performance was measured. The measurement items are average number of the CVS transactions per unit of time and average response time per unit of time under below conditions.

3.1.1. Effect of certificate, CRL/ARL and certification path cache

In the experiment, the data of the 500 transactions were obtained on the condition that 5 transactions were always executed simultaneously on a CVS. And the average value of the result was calculated by the middle 300 transactions (from 101st to 400th) within 500 transactions. The certification path length was 4.

- A-1 Proceeding without any cache
- A-2 Proceeding with certificates and CRL/ARLs cache
- A-3 Proceeding with certificates, CRL/ARLs and certification paths

3.1.2. Effect of revocation table

In the experiment, the data of the 500 transactions were obtained on the condition that 5 transactions were always executed simultaneously on a CVS. And the average value of the result was calculated with the middle 300 transactions (from 101st to 400th) within 500 transactions. The certification path length was 2. The number of revocation record of the CRL was 20,000.

- B-1 Proceeding without revocation table
- B-2 Proceeding with revocation tables

3.2. Experimental environment

The experimental environment is shown in figure 5. The client, the CVS, the directory server, and the CA were connected to the same router. And the network speed was 100Mbps.

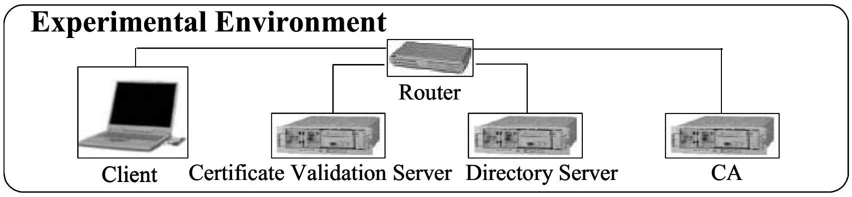


Figure 5. Experimental environment

The machines used in experimental environment are shown below.

Client

CPU	Pentium® III 850MHz
Memory	762MB
Operating System	Windows® XP operating system

Certificate Validation Server (CVS)

CPU	Ultra SPARK™ II 500MHz
Memory	4GB
Operating System	Solaris®

Directory server

CPU	Pentium® III 700MHz
Memory	762MB
Operating System	Windows® 2000 operating system

4. Result and discussion

4.1. Effect of cache of certificate, CRL/ARL and certification path

Table 1 shows the result of experiment executed on condition A1~A3.

Table 1. Measurement result of cache effect

	Average response time (sec)	Average performance (umber of transactions/sec)
A-1 Proceeding without any cache	1.54	3.17
A-2 Proceeding with certificate and CRL/ARL cache	0.86	5.57
A-3 Proceeding with certificate, CRL/ARL and certification path	0.75	6.56

Comparing the results of A-1 and A-2, the average of response time was decreased to 1/1.8 times and the average of performance was increased to 1.8 times, by effect of the cache of certificates and CRL/ARLs. On the other hand, comparing the results of A-2 and A-3, the average of response time was decreased to 1/1.1 times and the average of performance was increased to 1.2 times, by effect of the cache of certification paths. In total, the average of response time was decreased to 1/2.1 times and the average of performance was increased to 2.2 times, by effect of the two cache algorithms.

As described above, the speed-up effect of the certificate validation processing was obtained by the cache algorithm of certificates, CRL/ARLs and certification paths. And it was recognized that the cache of certificates and CRL/ARLs was more effective than the cache of certification paths in this experimental environment. Then, it is expected that many resources were consumed to gather certificates and CRL/ARLs when processing of certificate validation in the experimental environment.

In general, the effect of these caches appears when certificates and CRL/ARLs which the certificate validation function has obtained and used once were reused. Therefore, it is expect that this effect is limited in the environment such as client that cached certificates and CRL/ARLs are not reused many times. On the other hand, the effect of cache is expected for the servers executing large amount of transactions by using the same certificates and CRL/ARLs because the hitting rate of cache would be increased.

4.2. Effect of revocation table

Table 2 shows the result of experiment executed on condition B1 and B2.

Comparing the results of B-1 and B-2, the average of response time was decreased to 1/32 times and the average of performance was increased to 30 times, by effect of the building revocation table. The reason B-2 processing using large CRL was faster than A-3 processing is that the certification path was shorter than that of A-3.

Table 2. Measurement result of effect of revocation table

	Average response time (sec)	Average performance (number of transactions/sec)

B-1 Proceeding without revocation table	19.7	0.50
B-2 Proceeding with revocation table	0.61	15

As described above, the building of revocation table reduced the average processing time and improved the average processing ability, and thus realized the speed-up of certificate validation processing. It is considered because the processing to search records in CRL is reduced by sorting information beforehand.

In general, the certificate validation processing ability can be improved by building validation table under the environment of issuing large CRL such as large-scale certification infrastructure.

5. Conclusion

The speed-up of certificate validation processing was realized by caching certificates, CRL/ARLs, and certification paths. The above caching function is considered effective for improving processing speed especially for the server executing large amount of validation processing.

In addition, the validation speed was significantly improved by using sorted records of the CRL where the certificate validation was executed by using CRL including 20,000 records. The processing ability is expected to be improved by making CRL records for certificate validation processing under the environment of issuing large CRL such as large-scale certification infrastructure.

Refference

[1] ITU-T Recommendation X.509 "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks "(2000)
[2] A. Frier, P. Karlton and P. Kocher "The SSL 3.0 Protocol" "http://wp.netscape.com/eng/ssl3/ssl-toc.html" (1996)
[3] T. Dierks and C. Allen: RFC 2246:"The TLS Protocol Version 1.0" (1999)

[4] United State Department of Justice “Electronic Signatures in Global and National Commerce Act”, Federal low (USA)
[5] Japanese Ministry of Internal Affairs and Communications “Law on Electronic Signatures and Certification Services”, Low No 102 of 2000 (Japan)
[6] Chief Information Office Council “Federal Bridge Certification Authority” http://www.cio.gov/fbca/
[7] Japanese Ministry of Internal Affairs and Communications, “Governmental Publickey Infrastructure specifications for mutual operation” http://www.gpki.go.jp/session/CompatibilitySpecifications.pdf (2001)
[8] “the Public Certification Service for Individuals” http://www.jpki.go.jp/
[9] T. Fujishiro, T. Kaji, S. Hane, Y. Kumagai and S. Tezuka „Development of Certificate Validation Service” The IEICE Transactions on Information and Systems, PT.1 (Japanese Edition), Vol. J87-D-I No.8, 833 (204)
[10] R. Housley, W. Ford, W Polk and D. Solo: RFC 3280: "Internet X.509 public key infrastructure certificate and Certificate Revocation List (CRL) profile" (2002)

[11] A. Malpani, S. Galperin, M. Myers, R. Ankney and C. Adams: RFC 2560: "X.509 Internet public key infrastructure online certificate status protocol - ocsp" (1999)
[12] A. Malpani、 R. Housley and T. Freeman: IETF Internet drafts: "Simple Certificate Validation Protocol (SCVP)" "draft-ietf-pkix-scvp-16.txt" 2004(work in progress)

Trademarks

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

A Lightweight Delegated Privileges Revocation Scheme Based on Coding

M. Francisca Hinarejos and Jordi Forné[1]
Telematics Engineering – Technical University of Catalonia – Spain

Abstract. Public Key Infrastructure (PKI) and Privilege Management Infrastructure (PMI) can respectively be used to support authentication and authorization in distributed scenarios. Certificate path processing is a critical issue in both infrastructures, because it requires several costly processes, such as certificate path discovery, validation of the digital signature and checking the revocation status of each certificate. The problem becomes much more complex when using delegation of privileges in a PMI, because in this case the length of the delegation paths can be high. In this paper, we propose a revocation scheme devised to reduce the communication and computational overhead for certificate status checking in delegation paths. This goal is achieved by using suitable coding techniques.

Keywords. Certificates Revocation, Delegation paths, Authorization, PMI.

Introduction

Authentication is the assurance that the communicating entity is the one which it claims to be, while authorization is the process to verify that an entity has sufficient rights to access the requested resources. Traditional access control relying on authentication and enumeration of subjects in Access Control Lists (ACLs) needs to keep the ACL consistent and up-to-date. This process is difficult and presents important scalability problems in distributed environments, especially when facing multi-domain trust and policies.

In order to overcome these problems, the ITU-T defined in [1] a new type of certificate, the attribute certificate (AC). A Public Key Certificate (PKC) binds an identity with a public key, whereas an AC binds an identity with a set of attributes. A Privilege Management Infrastructure (PMI) is a collection of Attribute Certificates (ACs), with its issuing Attribute Authorities (AAs), subjects, reliant parties and repositories. Through the PKIX group [2], the IETF [3] is also adapting attribute certificates for authorization in the Internet.

A PMI allows the following:

1) Support for distributed (role-based) access control: With ACs, it is possible to bind an identifier and a set of attributes that can describe rights or privileges, rendering ACLs unnecessary.

[1] Engineering Telematics Department. Polytechnic University of Catalonia (UPC) Jordi Girona 1-3. Campus Nord, C3. 08034 Barcelona. E-mail: {mfcampos, jforne}@entel.upc.edu.

2) Support for delegation of rights: Subjects can delegate their permissions to other subjects with no interaction with the authority during the delegation, allowing more decentralized authorization schemes.

A general chain of certificates is formed of *n* certificates which can be represented by form of a list $\{Cert_1,...,Cert_n\}$. We define each certificate in the list as follows:

1) The first certificate is $Cert_1$ and corresponds to the SOA (Source of Authority) in the PMI. The ITU-T defines SOA as "an Attribute Authority that a privilege verifier for a particular resource trusts as the ultimate authority to assign a set of privileges".
2) The last certificate in the chain is $Cert_n$ and is a user certificate or a certificate to be validated by a privilege verifier. We have referred to this certificate as the 'target certificate'.
3) Any intermediate certificate $Cert_i$ where $1<i<n$ represents a certificate issued to an authority. This authority can be either a certification authority in PKI or an attribute authority in PMI.

PMI based authorization implies the verification of a single certificate or a complete attribute certificate chain (which is called a delegation path). The validation process includes the revocation status checking, a costly procedure with important scalability problems [4]. In fact, the complexity is much higher than certificate status checking in a PKI, because in a PMI the length of the delegation path can be longer than the certification path (or chain of public key certificates).

In this paper, we propose a scheme that facilitates the status checking of a delegation path for a PMI. A revocation policy in cascade is used to reduce the complexity of the status checking when validating a delegation path. The main idea is quite simple: when certificate C is revoked, all the attribute certificates issued by the entity C (or by any other entity that derives its privileges from C) are revoked as well. This scheme guarantees that if the last certificate of the delegation path is not revoked, then all the certificates in the delegation path are similarly not revoked. This approach is suitable for scenarios where the certificate revocation is achieved because the holder of the privileges has used his or her privileges in a fraudulent manner and, therefore, the entities cannot trust all certificates issued by the dishonest holder.

The proposed scheme is based on a suitable coding of the identifier associated to an attribute certificate, which may be its serial number. The encoding scheme allows the establishment of both the delegation path (from a target certificate to the root certificate) and the certificates issued from the identified certificate. This fact allows a easy implementation of a revocation policy in cascade in an implicit way. In this way, this scheme reduces both the volume of revocation information and the burden on the validation process of the delegation path since it is only necessary to check the revocation status of the target certificate.

The rest of the paper is structured as follows. In section 1, several features about certificate path processing are presented. Section 2 presents the background for revocation of privileges, including policies and status checking protocols. Issues related to cascade revocation in delegation paths are addressed in section 3. In section 4, we propose the new revocation scheme. Finally, in section 5 we draw our conclusions and ongoing work.

1. X.509 Attribute Certificates

Several systems use PKCs to manage the decisions based on access control. However, sometimes the information provided on the PKC is not detailed enough to warrant decision-making about access control. For this reason, new mechanisms based on another type of feature are necessary, such as roles[2] or user privileges. A PKC could convey this type of information; however this is not a suitable solution due to the dynamic nature of the privileges:

- The validity period of a privilege or attribute can be quite low (minutes, hours, etc.) compared to the PKC validity period (months or years). If the attributes are included in a PKC, it would incur certificate revocation when some of its attributes expire, and it is known that certificate revocation is a costly process [4].
- The certification authority is a source of authority for identities, but it does not have to be the source of authority for privileges. For example, an authority as administrator has the authority to verify the user identity, but it does not have the authority to grant privileges on the resources of a company. The administrator of the resources within the company could be the entity with authority to grant privileges on the resources, such as: to print, to sign files, to place orders with suppliers, and so on.
- The PKC does not provide several other features required for the authorization processes as the delegation: transfer privileges or a subset of these to another user.

To solve these problems, the ITU-T included the attribute certificate (AC) and its framework (PMI – Privilege Management Infrastructure) into the X.509 Recommendation [1]. PMI can be defined as[3]: the set of hardware, software, persons, policies and procedures needed to create, manage, distribute and revoke attribute certificates (AC). An attribute certificate is defined by the ITU-T as a data structure, digitally signed by an Attribute Authority, that binds some attribute values with identification information about its holder.

X.509 [1] defines two types of certificates chains: a) certification chains, which consist of the public key certificates, and b) delegation chains, which consist of the attribute certificates. The problem of validating the delegation chains is more complex than that of verifying the certification chains. This is due to the fact that verification of delegation chains involves at least the validation of the certification chain linked to each AC in the delegation path, see Figure 1; and the delegation chains can be longer than the certification chains.

Figure 1 depicts the relationship between the different certificates in the delegation path. The validation involves the following steps:

[2] Reflect the privileges or permissions that belong to a user on a resource.

[3] Defined by the IETF in [2].

- Obtain the PKC bound to the presented AC.
- Obtain and validate the certification path associated to the PKC.
- Verify the AC signature. The privilege verifiers obtain the PKC of the attribute authority which conveys the public key necessary to achieve this process.
- Obtain the next AC in the path. The two last steps must be repeated until the AC SoA is achieved. In the worst case scenario, each certification path involved could belong to different CA domains [5]. This fact must be taken into account because it increases the complexity and the burden in the system.

In the process described below, if any certificate in the chain has been revoked, either the identity or attribute certificate, the complete path is already not valid. Therefore, if the revoked certificate gets further from the certificate in order to validate, the result is that the unnecessary cost on the system grows.

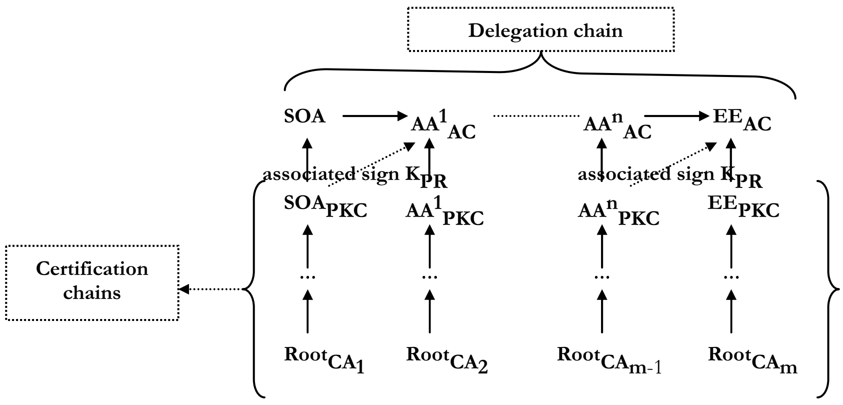


Figure 1. Certificates chains involved in the delegation general mode defined by the ITU-T in the X.509 Recommendation [1].

In the following sections we are going to focus on the revocation policies. We will describe a scheme to achieve the implementation of a suitable revocation policy, which will reduce the burden on the process of a certificates status checking.

2. Revocation of privileges

An attribute certificate must be revoked when some of its associated privileges are no longer valid. This can be for several reasons, such as fraudulent use of the privileges by the user. When an attribute certificate is revoked, the other entities must be aware that the revocation has occurred.

In this section we explain two processes related to the privileges revocation: 1) the steps to revoke a privilege (or a certificate carrying the privilege), and 2) the mechanisms used to check the certificate revocation status.

2.1. Revocation policies

It is often necessary to delegate privileges or user rights from one user to another user in a system. This process must be performed by the authorized entity. The revocation of the delegated privileges should be performed by authorized entities, such as: the privilege holder, the entity that delegated the privilege or another authorized party. A further issue to consider is how the revocation affects the rest of the privileges either directly or indirectly.

The authors in [6] identify three dimensions – resilience, propagation and dominance – showing the different scenarios at the time of revoking privileges. In fact, these dimensions can be combined to provide different categories. Below, we abstract the three dimensions:

- Resilience. This dimension refers to the persistence of the revocation over time: 1) the revocation via the deletion of the privilege. This revocation has a temporary effect because the privilege can be re-granted; and 2) the revocation via the granting of a negative privilege. Therefore, the negative privilege overrides the positive privilege until the negative privilege is deleted.
- Propagation. Unlike the resilience dimension with temporary effect, the propagation dimension has a spatial effect. That is to say, the way in which users are affected by the revocation of a certain privilege: 1) Local: Only affects the user whose privilege is revoked; and 2) Global: the revocation is in cascade. The revocation affects both the user and the same privilege granted to other entities by the entity that revokes the privilege.
- Dominance. Suppose that an entity A revokes the granted privilege to another entity B, but entity B still owns privileges granted by other entities. In this model two categories can be differentiated: 1) strong revocation if the entities were granted privileges from a certificates path, and the entity that revokes the privileges is contained in the same path, then this entity can revoke the grantees; 2) weak revocation, the different entities and the entity revoking the privilege are independent. In this case, only the privilege granted directly by the entity that revokes the privilege is revoked.

The classification presented is a conceptual classification. In fact, its application through X.509 attribute certificates can become unsuitable. The rest of the paper considers the following assumptions:

1) When a user attribute certificate is revoked, the certificate is no longer valid.
2) The revocation has a cascade effect on the certificates which have their root in the revoked certificate.

However, the last point is a decision based on the current revocation policy. In other words, if the certificate is revoked because the holder has made a fraudulent use, the cascade revocation effect is achieved. It is necessary because the entities cannot trust all certificates issued by the dishonest user, although there are certificates issued in an authorized manner, it is difficult to distinguish between the certificates issued non-fraudulently.

2.2. Revocation Status Checking

Once a certificate (or privilege conveyed in a certificate) is revoked, there must exist a mechanism that allows for checking if the certificate is still valid or not. There are different revocation mechanisms that allow to retrieve the revocation information. In the vast majority of the revocation systems, end entities do not have a direct connection neither to the repositories nor to on-line servers. The distribution of the status information can be performed in two ways:

- Offline: the entities used for offline status data distribution are not TTPs (Trust Third Party). In this case the issuer pre-computes the status data and distributes it to the repositories, such as CRL [1]. The ITU-T has defined in [1] a new type of structure, the ACRL (Attribute Certificate Revocation List), which has the same structure than CRL, but that defines the list of the revoked attribute certificates.
- Online: the entities used for online status data distribution are TTPs. In this case the server offering the service provides the cryptographic evidence for the status data that it produces, such as OCSP [7].

The currently standardised status checking mechanisms [1,7] are focused on: 1) reducing the trust on the entity which distribute the revocation information [1], or 2) reducing the bandwidth used to disseminate the revocation information [7]. However, the complete list of the revoked certificates identifiers must be known.

The volume of revocation information and the number of updates is likely to increase in authorization systems based on attribute certificates. These features can be aggravated when the delegation process is allowed, so it is difficult to validate all certificates on the delegation path. This problem is further addressed in section 3.

3. Cascade revocation for delegation paths

In several scenarios privileges can be delegated from one entity to another one. These delegations can be depicted by hierarchical structures. Figure 2 depicts a sample of a hierarchical relation of delegating privileges between different entities. Let us suppose that the entity A holds a set of four privileges {1,2,3,4} and delegates a subset of her privileges to B {1,2}, and the privileges subset {1,3,4} to the entity C. Similarly, the entity C delegates the privileges {1,4} to entity D. Finally, the entity D delegates the privilege {4} to the entity E. Delegation is only possible when the entities hold authority to delegate the privileges. In fact, the length of the delegation path can be limited by the entity that is source of authority of the privileges. In other words, if the entity A fix the limit of the delegation path to two, then the entity E can not delegate the privilege {4} to any one.

Sometimes it is necessary to revoke a privilege to an entity due to, for example, a fraudulent use of the privileges by the user. If A revokes the privilege 4 to C, the same privilege granted by C to other entities must be not valid too. So, the privilege 4 granted to both D and E entity is no longer valid. This situation corresponds with the global propagation dimension explained in [6], and this will be the focus of our work.

Let us suppose now that the privileges are conveyed in digital certificates and the privilege 4 of the entity C is revoked (see figure 3). A verifier must get the certificates path from A to E {A,C,D,E} to verify if entity E holds the privilege (this process is known as path discovery). The verifier must go on to check if any certificate in the path is revoked. The path discovery is a costly process [8, 5] and it must be carried out, although any certificate in the path has been previously revoked. This is because the verifier does not know the identifiers of the certificates beforehand. Therefore, it cannot request information about the status of the certificates in the path.

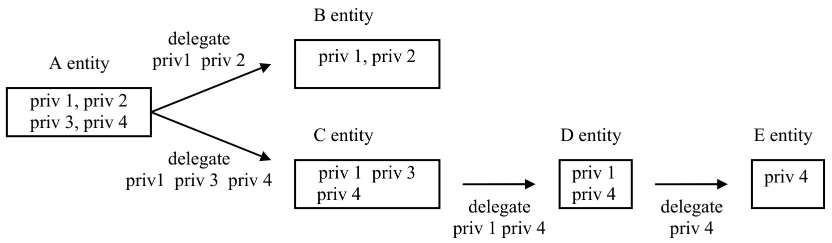


Figure 2. Privileges delegation process. Relationship between the entities and the delegated privileges.

An easy solution consists of including the identifiers of the delegation path in each issued certificate (from the root to the issuer of the target certificate). This information may be included in an extension field of the target certificate. For instance, in figure 3, E's certificate must include the identifiers of certificates A, C and D. When an entity tries to validate the E's certificate, it obtains the identifiers from the extension of the E's certificate. Then, the verifier requests the revocation status of each obtained identifier. If any certificate is revoked, it is not necessary to achieve the path discovery process. However, it is necessary to check the status of each certificate.

Another solution is updating the certificates dependencies when the revocation is generated. That is to say, if a certificate is revoked, then the dependant's certificates are also revoked. In figure 3 when $Cert_C$ is revoked, then certificates $Cert_D$ and $Cert_E$ are revoked as well. The solution of including the identifiers of the delegation path in a certificate is unsuitable since the included information correspond to the certificates from the root to the target certificate. However, the certificates from the target certificate to any dependant certificate are required. Unfortunately, acquiring this knowledge is not an easy task.

To overcome this drawback, it could be necessary to trace the delegation path. When the privileges of the entity C are not longer valid, certificates $Cert_C$, $Cert_D$ and $Cert_F$ are marked as revoked as well. In this case, it is only necessary to validate the revocation status of the target certificate. Therefore the revocation status of $Cert_F$

reflects the revocation status of the complete delegation path. This procedure allows the certificates identifiers in the delegation path to be supplied, as well as helping in the path discovery process.

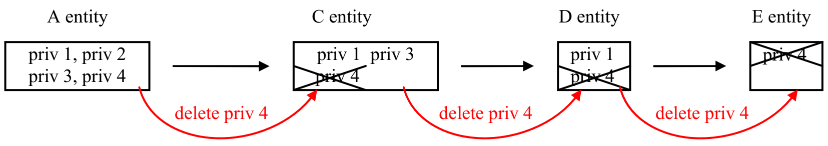


Figure 3. Privileges deletion process. Process to revoke the privileges delegated to entities with hierarchical dependencies.

4. Proposed revocation scheme

We propose a revocation scheme based on coding[4] the identifiers of attribute certificates through suitable code-words. This particular coding led us to easily implement a cascade revocation policy. This simple technique has the following advantages: 1) easier delegation path discovery (as it was introduced in [9]), and 2) reducing complexity in certificate status checking (as it will be explained). The latter is true because it is not necessary to verify the revocation status for each certificate on the delegation path. Furthermore the procedures to achieve these goals are explained.

4.1. Tree coding

The certificate dependencies can be depicted through a hierarchical tree[5]. Figure 4 depicts the certificates dependencies through a logical tree: 1) each node represents a user attribute certificate and each branch from the root to the target certificate represents the delegation path, 2) if the node 2 is revoked, the nodes 4, 5, 6 and 7 must not be longer valid due to the dependencies among the nodes.

If a different identifier is given to each node, the certificate can be identified in a univocal way in the complete tree. Furthermore, if that identifier is generated from the identifier of the parent node, we achieve a scheme that allows both to represent the certificates dependencies and to identify the certificates in a univocal way.

We must choose the suitable coding which must carry out the features previously exposed, that is:

- The certificate identifier must be unique under each attribute authority.
- The identifier must provide information to ascertain the certificates involved in delegation path (we denote this fact as down-up knowledge).
- The identifier must provide information to ascertain the certificates issued from a target certificate (we denote this fact as up-down knowledge).

[4] For further information about coding and compression techniques see [14].

[5] The relations in the delegation process can lead to general graphs; however this issue is out of the scope of the paper.

Therefore it is possible to know the revocation of the dependant certificates in an implicit way, without obtaining the identifier of the dependant certificates.

There are many coding techniques that fulfill the previous requirements. For the sate of simplicity, in this paper, we adopt the approach from [10] to construct a binary tree from a multi-way tree (the same adopted in [9] to carry out the certificate path discovery). In this case, each child codeword is generated by concatenating the certificate parent codeword with a new codeword. That is to say:

- The first child is generated by concatenating the parent codeword with a 0.
- The codeword of the rest children at the same level are generated by concatenating the codeword of the first child with a number of 1's equal to n-1, where n is the child number under the direct domain of the parent node.

This coding allows to convert a multi-way tree to a binary coding tree. Figure 5 depicts a possible logical procedure to generate a codeword as above described.

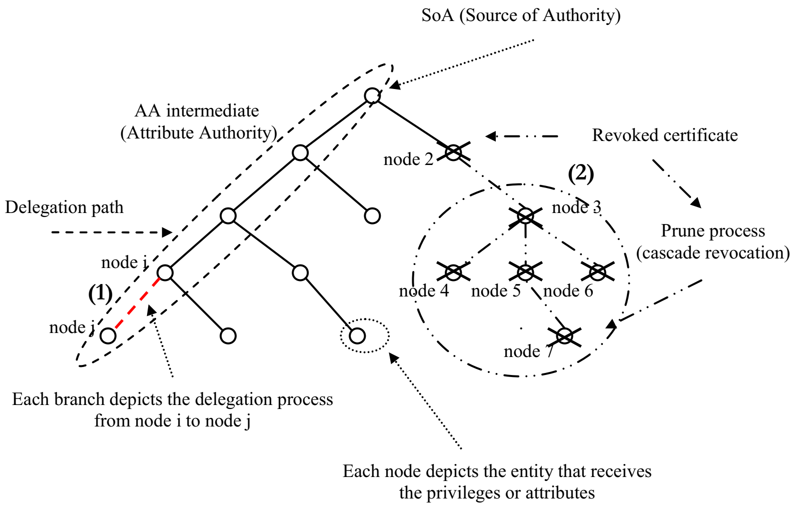

SoA (Source of Authority)
AA intermediate (Attribute Authority)
Revoked certificate
node 2
(2)
node 3
Delegation path
Prune process (cascade revocation)
node i
node 4
node 5
node 6
(1)
node j
node 7
Each branch depicts the delegation process from node i to node j
Each node depicts the entity that receives the privileges or attributes

Figure 4. Logical representation of the dependencies among the privileges certificates.

This type of coding allows the generation of a different codeword for each node of the tree, therefore, each certificate identifier is unique. Furthermore, the prefix codeword gives implicit knowledge of the delegation path, from the root node to the target certificate (*down-up knowledge*). However, the most important feature for our revocation scheme is that we can know the dependents certificates from the goal certificate in an implicit way (*up-down knowledge*). For instance, in Figure 6 the codeword D→X10 allow to get the certificates identifier that forms the delegation path, C(X1) and A (X) since the D codeword contains the necessary information. On the other hand, any certificate with the codeword structure "X10||any codeword" is dependant of codeword "X10", where "||" denotes concatenation.

```
function codeWordGeneration ( $cw_{parent}$ , $pos_{child}$ )
{
    $cw_{child} = cw_{parent} \| 0$
    // if the first child
    if ( $pos_{child}$ == 0) then
        return ( $cw_{child}$ );
    for (int i=0;i< $pos_{child}$ ;i++)
    {
        $cw_{child} = cw_{child} \| 1$;
    }
    return ( $cw_{child}$ );
}
```

The first child is generated by the concatenation of the codeword parent with one 0

The rest of children are generated by the concatenation of the codeword of the first child with n-1 1, where n is the child number under the direct domain of the parent node

Figure 5. Pseudo-code algorithm to generate the codewords nodes representing the attribute certificates.

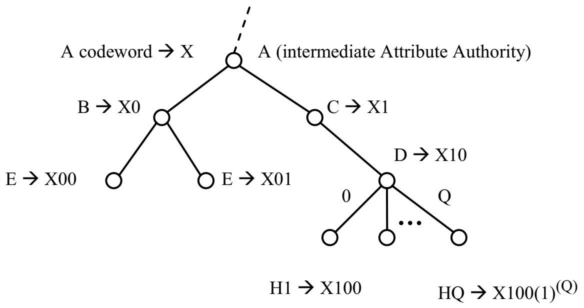


Figure 6. Coding tree representing the attribute certificates.

The tree depicted in figure 4 should be maintained by a centralized entity, as explained in section 3. However, the special coding of the attribute certificate identifier makes this process unnecessary, since the all of the information is implicitly included in the certificate identifier, and therefore, in the certificate itself.

The same idea introduced in [9] is adopted, but unlike [9] where the coding of the identifiers is used for down-up knowledge, we use identifiers coding for up-down knowledge.

4.2. Codeword in attribute certificates

The attribute certificate must include its identifier. In our scheme this identifier is the associated codeword. Inside the same attribute authority each certificate is differentiated from the other certificates by the serial number. Therefore, the serial number field is the most suitable one to convey the codeword (see figure 7).

The codeword identifying a certificate has two parts (see figure 7): 1) the prefix depicts the codeword of the parent certificate; and 2) the codeword that identify a certificate among the other certificates under the same parent authority (we have detonated this part as child-code).

The length of the codeword can be long if the number of certificates issued by the same authority is high. We can divide the codeword into two parts: prefix-code and child-code. In a certificate there are a field to identify the issuer, so we can use this field to convey the prefix-code, and the serial number field to convey the child-code. This technique allows the increase of the number of certificates issued in each level.

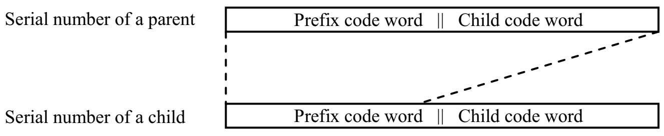

Serial number of a parent
Prefix code word || Child code word
Serial number of a child
Prefix code word || Child code word

Figure 7. Coding the serial number of the attribute certificates.

The coding representing the certificates delegation path could be included in an extension of each certificate. However, the existing protocols could not be reused because they are based upon the certificate serial number. Therefore, the implementation would require more operational changes.

4.3. Run Length Encoding of certificate identifiers

In the section 4.2 we commented that the length of the codeword can be long either if the number of certificates issued by a same authority is high, or if the depth of the multi-way tree is high as well. In other words, the maxim number of bits necessary to represent a codeword is $qloq_q n$ where q denotes the maximum number of certificates issued by the same attribute authority, and $\log_q n$ denotes the depth of the tree. A source coding algorithm can be used to reduce the volume of information to include in the certificate.

The codeword generated in section 4.1 follows a well defined structure. This structure can be generalized for each level as a sequence $\{01...1\} = \{01^q\}$ where 1^q denotes a sequence of q ones. Run Length Encoding (RLE) is a simple and popular data compression algorithm and it is very useful to compress this kind of sequences [11]. It is based on the idea to replace a long sequence of the same symbol by a shorter sequence. The sequence of length l of a repeated symbol 's' is replaced by a shorter sequence, usually containing one or more symbols of 's', a length information and sometimes an escape symbol. For example, if an attribute authority issues 10,000 certificates, the last codeword generated will contain about 10,000 bits (approximately 1,250 bytes) plus the bits of the codeword parent. This information contains a high degree of redundancy and it reduces the viability of the revocation scheme. However, if the child codeword is represented as an integer indicating the number of ones, only 14 bits are necessary (approximately 2 bytes). By using this simple technique, the data to include on the certificate is reduced.

4.4. Generation of the revocation tree

There are two phases in the revocation process: 1) the request of revocation certificate, and 2) the mechanism to get the revocation information [1, 7, 12, 13]. The request of revocation process is similar to the existing methods. The main information contained in the request is the certificate identifier which is a codeword in our scheme. This allows us to reuse the existent protocols for revocation request [14].

We define *revocation tree* as the tree which contains the revocation information. When a revocation request is received, the revocation tree has to be updated according to the given policy. Firstly, it has to be verified if the codeword to be revoked is an internal node of the revocation tree[6]. If so, all the branches under the prefix have to be pruned. On the other hand, if the codeword does not exist, a new entry has to be created. This feature allows the reduction of the revocation information when the revocation involves more than one certificate.

Figure 8 (a) depicts a revocation tree of three certificates. In this instance, the certificate with codeword X1 is revoked. X1 is prefix of the codeword both X100 and X1011, therefore the codeword X1 reflects the codeword X100 and X1001 in an implicit way. Therefore the coding of the revocation tree reduces the volume of revocation information to store.

Every time an entity requires the checkup of the certificate status, only the status of the target certificate needs to be verified. In other words, the entity is not required to verify the status of each certificate in the delegation path, unlike the existing solutions where the verifier must get the complete path of certificates and check the status of each certificate.

4.5. Revocation status checking

The revocation status checking is quite direct. For instance, for verifying the status of the certificate identified by X1011 on the tree depicted by figure 8 (b), the following steps must be carried out:

- Obtaining each binary digit of the codeword starting from left to right.
- The first digit is the “1” therefore it is corresponded to the right branch of the tree.
- The node X1 is both a leaf (it is a revoked certificate) and a prefix of the codeword X1011, then the certificate X1011 is revoked in an implicit way. So, the status checking is finished.

[6] The leaf node or final node in the tree contains the complete information about the revocation certificates, but an internal node does not depict a revoked certificate unlike the coding tree (see figure 6) where each node depicts an issued certificate.

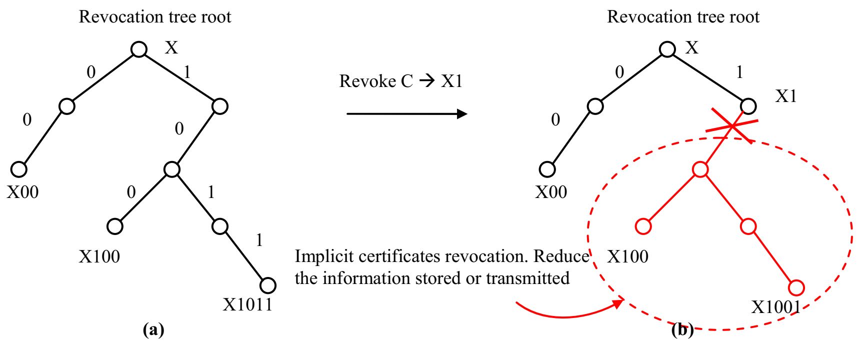


Figure 8. Revocation tree representation. (a) Sample tree of revoked certificates. (b) Effect of revoking a single certificate in the revocation tree.

The figure 9 depicts a pseudo-code algorithm which allows to check if an attribute certificate is revoked, following the steps explained above.

```
function verifyCWRevocation ( $cw_{certificate}$ )
{
     $cw_i = cw_{certificate}$ ;
     $l_{cw} = |cw_i|$ ;
     pointer = tree.root;
     for (int i=0;i< $l_{cw}$ ;i++)
     {
       if pointer.left.bit != $cw_i(i)$ || not_exist then
       {
             if pointer.right.bit != $cw_i(i)$ || not_exist then
                 return(not revoked);
             else if pointer.isleaf then
                 return(revoked);
             else
                 pointer = pointer.right;
       }
       else
       {
             if pointer.isleaf then
                 return(revoked);
             pointer = pointer.left;
       }
     }
}
```

Figure 9. Pseudo-code algorithm to check the revocation status of an attribute certificate.

5. Conclusions and Ongoing Work

The validation of the certificates delegation path is a critical issue in infrastructures based on digital certificates. This is because several costly processes are required, such as certificate path discovery and revocation status checking of each certificate in the delegation path.

As far as we know, all the approaches presented up-to-date require previous path discovery to obtain the identifiers of the certificates in the delegation path. These identifiers are necessary to check the revocation status of each certificate. This can be a costly process when the delegation path is long.

The proposed revocation scheme allows us to reduce the necessary knowledge of the certificates delegation path, at the moment of the validation. By using this scheme, to know if any certificate in the path is revoked, we only need to check the revocation status of the last certificate in the delegation path. This interesting property is achieved through the special coding of the certificate identifier. As an additional advantage, this approach makes the process of path discovery only necessary when the target certificate is not revoked.

The proposed revocation scheme reduces the communication and computational overhead of the certificate status checking in a complete delegation path. Therefore, our scheme is particularly suitable for scenarios where the end entity has limited resources - memory, computational capacity and bandwidth -, such as mobile devices.

A general operation of a revocation scheme based on certificates identifier coding is presented in this paper. The way in which this scheme improves the performance of the more common revocation mechanisms used to obtain the revocation information, such as ACRL and OCSP is part of our current research.

In order to achieve this goal two different parameters must be evaluated: the reduction of the bandwidth and the computational overhead. This improvement is direct as either the number of operations and the volume of send information is fixed with the length of the delegation path. However, the value can improve or worsen with the percentage of the certificate revocation at the different levels of the logical tree. Furthermore, the use of different coding techniques could improve the performance of the revocation scheme.

Acknowledgements

This work has been supported by the Spanish Research Council under the project ARPA (TIC2003-08184-C02-02) and the European Research Council under the project UBISEC (IST-FP6 506926). We also thank the *Departament d'Universitats, Recerca i Societat de la Informació* that supports Francisca Hinarejos's PhD work.

References

[1] ITU-T Recommendation X.509, Information technology – Open Systems Interconnection – The Directory: Public Key and Attribute Certificate Frameworks. 2000.

[2] RFC 2459. SPYRUS, W. Ford, VeriSign, W. Polk, NIST, D. Solo, Citicorp: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. January 1999.

[3] IETF RFC 3281, S. Farrell.R. Housley: An Internet Attribute Certificate Profile for Authorization. April 2002.

[4] J. Muñoz, J. Forné, O. Esparza, M. Soriano, and D. Jodra, Evaluation of Certificate Revocation Systems with a JAVA Test-Bed. DEXA 2003. Workshop on Trust and Privacy in Digital Business (TrustBus03), IEEE Computer Society.
[5] Steve Lloyd, PKI Forum: Understanding Certification Path Construction. White Paper. September 2002.
[6] Hagstrom A.; Jajodia S.; Parisi-Presicce F.; Wijesekera, D.: Revocations –a classification. Computer Security Foundations Workshop, 2001. Proceedings 14th IEEE, 11-13 June 2001. pp. 44 -58.
[7] RFC 2560. Myers M., Ankney R., Malpani A., Galperin S., and C. Adams: X.509 Internet Public Key Infrastructure – Online Certificate Status Protocol – OCSP. June 1999.
[8] RFC 3379. D. Pinkas, Bull, R. Housley, RSA Laboratorios: Delegated Path Validation and Delegated Path Discovery Protocol Requirements. September 2002.
[9] He Huang, Shyhtsun Felix Wu: An approach to certificate path discovery in mobile Ad Hoc networks. Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, October 2003, pp. 41-52, ISBN:1-58113-783-4.
[10] J. Stasko, J. Vitter: Pairing Heaps Experiments and Analysis. In Communications of the ACM, March 1987.
[11] M. Nelson: The Data compression book. M&T Books, New York, NY 1995. ISBN 1-55851-434-1.
[12] S. Micali: Efficient Certificate Revocation, Massachusetts Institute of Technology. Cambridge, MA, 1996.
[13] P. Kocher: On certificate revocation and validation. International Conference on Financial Cryptography (FC98). Lecture Notes in Computer Science, no. 1465, pp. 172-177, Feb. 1998.
[14] RFC 2510. C. Adams, Entrust Technologies, S. Farell, SSE: Internet X.509 Public Key Infrastructure Certificate Management Protocols. March 1999.

Cryptographic Applications

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Hiding Data Sources in P2P Networks [1]

Marek Klonowski, Mirosław Kutyłowski [2], and Bartłomiej Różański
Wrocław University of Technology
Institute of Mathematics and Computer Science

Abstract. We propose a peer-to-peer (P2P) architecture where the identity of nodes holding data remains hidden, but the information itself can be efficiently fetched. This architecture can be used to protect P2P networks against malicious attacks towards nodes holding important data. In particular, our protocol can be used for maintaining access to revocation lists, blacklists and similar data, which are of growing importance for modern P2P protocols.

Keywords. peer-to-peer, anonymous communication, blacklist

1. Introduction

Peer-to-peer (P2P) technology is an attractive basis for diverse application areas, not solely limited to file sharing environments. The advantages of P2P systems are, among others: robustness to node failures, high dynamics of nodes and scalability. Despite its features P2P paradigm had not yet drawn significant interest from commercial and business applications. The reason behind this situation is that the vast majority of modern P2P protocols implement no or very weak mechanisms to protect data flow between network nodes against adversaries monitoring the traffic. Usually, very little effort is required to track down source, destination and intermediate route of the packages. Therefore the adversary can shut down a given connection very easily, i.e. by performing a Denial of Service (DoS) attack. Moreover, if an adversary can eavesdrop connections between nodes, he can easily analyze traffic to deduce user preferences. This may lead to simple attacks against certain data and allow blocking of shared contents.

1.1. P2P based secure data servers

Obviously, the idea of implementing high security data servers in P2P networks is very appealing. Robustness to network dynamics and failures, as well as self-organization makes such servers very attractive and less expensive to operate – provided that they are protected against adversaries. Let us mention two application areas:

[1] Partially supported by the EU within the 6th Framework Programme under contract 001907 (DELIS)

[2] Corresponding author: Institute of Mathematics and Computer Science, Wrocław University of Technology, Wybrzeże Wyspiańskiego 27 PL- 50-370 Wrocław, e-mail: miroslaw.kutylowski@pwr.wroc.pl

P2P infrastructure for revocation lists Implementing Public Key Infrastructure through P2P networks is an appealing idea [1]. One of the major problems in PKI infrastructures is to provide reliable sources of certificate revocation lists. In a typical client-server architecture there is a server (or servers) holding such data. All participants of the protocol are aware of the place of storing such a sensitive data. Therefore, all over the time it requires strong protection against extensive attacks. Our experience shows that such attacks performed by skilled adversaries are not just hypothetical, but indeed very real and dangerous. Hence, dedicated PKI servers (resistant against such attacks) become very expensive and form substantial cost of applying X.509 PKI infrastructures.

A distributed system for revocation services would alleviate problems with hardware and communication failures. An example of such a system is Cornell Online Certification Authority (COCA) [2] based on a pool of P2P servers. Usual difficulty in this case lies in maintaining information consistency – which becomes non-trivial issue in large distributed networks (COCA is based on the concept of a *quorum*). Obviously, much easier solution would be to store data at just one or very few nodes. These few data holders would be much better secured, if they remained hidden from adversaries. At first glance, this may seem to be an infeasible task, since we also want a (seemingly) contradictory feature, i. e. efficient fetching of data. As we show further in this paper, these two features can be conciliated in a protocol which forms a solid building-block for modern P2P architectures.

Blacklists in a P2P network Selfish and unfair peer behavior is one of most significant problems of P2P networks nowadays. Lack of proper incentives may become an acute issue for commercial usage of P2P networks: commercial users may drop altruistic attitude and become selfish – they would use services intensively but provide very poor service. The way to solve this problem is to somehow enforce fair behavior.

In order to cope with this problem many trust mechanisms have been designed (see for instance [3]). The general idea is that a peer behaving correctly collects credentials. This this is a white-list approach. The same goal can be achieved with so called blacklists, where records of selfish peer behavior are stored and can afterwards be checked by other peers. The obvious problem with such blacklists is that they are stored at some network node(s) which is known to everyone. Hence, powerful but dishonest parties need little effort to block unfavorable information. Obviously, no security can be guaranteed here as long as the blacklist are stored at publicly known and accessible nodes.

1.2. Previous systems

Different aspects of secure access in P2P networks have been studied due to vulnerability of P2P networks to DoS attacks. A lot of work have been done in order to provide anonymous access to network resources (so this is the opposite goal: data sources are known, and the goal is to hide who is fetching them). The systems like CROWDS and Tarzan [6] were proposed with this goal in mind.

TURTLE system [4] proposes to use an overlay network based on trusted connections between *friends*. DoS attack becomes hard in such a system, since communication is restricted to friends. Fetching data is less efficient, it is implemented through a search in the network.

1.3. Proposed architecture

We present an architecture for hiding data sources in P2P networks, so that high protection against attacks towards network contents can be achieved. Our solution is based on universal re-encryption technique as well as dynamic data access structures. We show that our protocol is resistant to the adversary who can tap nodes and monitor their traffic.

1.4. Paper organization

In Section 2 we present message encoding methods used in our protocol. In Section 3 we describe the main protocol. In Section 4 we present an analysis of an adaptive attack performed by adversaries trying to detect data sources.

2. Encoding schemes

In this section we present cryptographic tools which will be used further in this paper to provide secure communication. We first describe *Onion Routing* protocol that allows anonymous communication over large scale networks using *anonymity paths*. Then *Universal Re-Encryption* (URE for short) is presented – a technique that allows partial decryption of ciphertexts. The combination of both methods – *URE-Onions*, which provide secure communication and protection against repetitive attacks, as well as allows partial ciphertext decryption is presented further in this section. Finally, a special type of URE-Onion called *Navigator* is presented – an URE-Onion with one element encoding 1. This allows “injection" of messages through nodes on the anonymity path in a secure way, i.e. no other node except the very recipient is aware of message transmission.

2.1. Onions

So-called *onions* are commonly used as a building-block for anonymous communication protocols in large scale networks [7,8,9]. Let us recall the basic mechanism. Each node in the network has a pair of keys. The public keys are widely known. To send a message m to node D, a server S chooses some random path of intermediate nodes, say $J_1, \ldots, J_\lambda$, called *anonymity path*. Let Enc_X denote a ciphertext of m computed with the public key of X. We assume that encryption scheme Enc is probabilistic. The onion encoding m has the following form (for simplicity we omit some less relevant details):

$$\mathrm{Enc}_{J_1}(\mathrm{Enc}_{J_2}(\ldots(\mathrm{Enc}_{J_\lambda}(\mathrm{Enc}_D(m), D), J_\lambda)\ldots), J_3), J_2) \ .$$

At the beginning this onion is sent by S to J_1. Node J_1 decrypts this ciphertext. It obtains a plaintext consisting of two parts: the second part is J_2 and the first part is the onion with one encryption layer “peeled off”:

$$\mathrm{Enc}_{J_2}(\ldots(\mathrm{Enc}_{J_\lambda}(\mathrm{Enc}_D(M), D), J_\lambda)\ldots), J_3) \ .$$

Now J_1 sends this ciphertext to J_2. The nodes $J_2, \ldots, J_\lambda$ perform similar operations, and the onion is subsequently peeled off until it finally reaches D. It can be easily seen that peeling off a single layer changes the onion thoroughly. Therefore, if two onions meet at the same node, they will be indistinguishable after leaving it (hence the correlation between node input and output is hidden from adversary). This kind of event is often called a *conflict*. In fact, anonymity of protocols based on onions is closely related to conflicts. Rigid mathematical analysis of protocols based on onions in case of passive adversary is given in [10,11,12].

2.2. Universal Re-Encryption

Let us recall El-Gamal encryption scheme: let G be a cyclic group of order q, for which discrete logarithm problem is hard. Let g be a generator of G. A private key is a random $x < q$, and the corresponding public key is $y = g^x$. A message m is encrypted in the following way: first some k, $0 < k \leq p-1$, is chosen uniformly at random. Then we put $r := g^k$ and $s := m \cdot y^k$. The pair (s, r) is the ciphertext of m.

El-Gamal cryptosystem has an interesting feature: everyone can re-encrypt ciphertext (α, β) so that the relation between (α, β) and a new ciphertext (α', β') is hidden for every party which does not know the private decryption key. Namely, if y is the public key used for ciphertext creation, one can choose some random k' and compute $\alpha' := \alpha \cdot y^{k'}$, $\beta' := \beta \cdot g^{k'}$. It is easy to see that the resulting pair (α', β') is a valid ciphertext of the same message. Golle et al. [13] presented a slightly modified El-Gamal scheme for which re-encryption of a message does not even require knowledge of the public key. It is called *universal re-encryption* (*URE* for short). In their scheme the ciphertext of message m (called *URE-ciphertext* of m) has the following form:

$$(\alpha_0, \beta_0; \alpha_1, \beta_1) = \left(m \cdot y^{k_0}, g^{k_0}; y^{k_1}, g^{k_1}\right) ,$$

where k_0 and k_1 are picked uniformly at random. This is obviously a pair of El-Gamal ciphertexts of messages m and 1 respectively. Decryption of this ciphertext is performed just as in the case of the original El-Gamal scheme: $m_0 := \alpha_0/\beta_0^x$, $m_1 := \alpha_1/\beta_1^x$, and the m_0 is accepted as a valid plaintext, iff $m_1 = 1$.
For re-encryption of $(\alpha_0, \beta_0; \alpha_1, \beta_1)$ random values k_0' and k_1' are chosen. New, re-encrypted ciphertext takes the following form:

$$\left(\alpha_0 \cdot \alpha_1^{k_0'}, \beta_0 \cdot \beta_1^{k_0'}; \alpha_1^{k_1'}, \beta_1^{k_1'}\right) .$$

2.3. Enforcing partial decryption

It is easy to enforce decryption of a message by some set of nodes [14]. Namely, the ciphertext has to have the following form:

$$E_{x_1,x_2,\ldots,x_\lambda}(m) = \left(m \cdot (y_1 y_2 \ldots y_k)^{k_0}, g^{k_0}; (y_1 y_2 \ldots y_k)^{k_1}, g^{k_1}\right)$$

where $y_1, y_2, \ldots, y_k$ are public keys of nodes which have to process the ciphertext, and $x_1, x_2, \ldots, x_k$ are their corresponding private keys. Hence,

$$E_{x_1,x_2,\ldots,x_\lambda}(m) = \left(m \cdot g^{k_0 \sum_{i=1}^{\lambda} x_i}, g^{k_0}; g^{k_1 \sum_{i=1}^{\lambda} x_i}, g^{k_1}\right)$$

is a ciphertext with decryption key $\sum_{i=1}^{\lambda} x_i$ and it can be re-encrypted. Moreover, it can also be partially decrypted with key x_1:

$$E_{x_2,\ldots,x_\lambda}(m) = \left(\frac{\alpha_0}{\beta_0^{x_1}}, \beta_0; \frac{\alpha_1}{\beta_1^{x_1}}, \beta_1\right) .$$

After partial decryption we get a URE-ciphertext for decryption key $\sum_{i=2}^{\lambda} x_i$. So in order to retrieve m the ciphertext must be decrypted with private keys x_1, …, x_k (however, not necessarily in this order).

2.4. URE-Onions

Unfortunately, the encryption method used for construction of onions as presented above, is cumbersome in some situations. For instance, it is impossible to manipulate internal layers without peeling off the onion. This makes it harder to prevent attacks such as repetitive attack. The problem can be alleviated with URE-ciphertexts [14]. Namely, let $E_x(m)$ denote an URE-ciphertext of m with decryption key x. As for the regular onion protocol, a path of nodes $J_1, J_2, \ldots, J_\lambda$ is chosen at random. Then a modified onion is built from λ ciphertexts, called *blocks*. The ith block, for $1 \leq i \leq \lambda - 1$ has the following form:

$$E_{x_{J_1}+\cdots+x_{J_j}}(J_{i+1}) .$$

and an additional, last block is computed as:

$$E_{x_{J_1}+\cdots+x_{J_\lambda}}(m) .$$

Alternatively, instead of J_{i+1} a block may contain any (random) identifier that can be understood as the address of J_{i+1} by he server J_i – in fact we apply this scheme in this way. The most important new feature of this encoding scheme is that we deviate from the encapsulation idea – messages for different routing steps are included in separate ciphertexts. Additionally, these ciphertexts are supposed to be permuted at random when the onion is processed.

Routing the onions First, a modified onion is sent to node J_1. When J_1 receives an onion, it partially decrypts and re-encrypts all its blocks. During decryption phase each block $(\alpha_0, \beta_0; \alpha_1, \beta_1)$ is replaced by

$$\left(\frac{\alpha_0}{(\beta_0)^{x_j}}, \beta_0; \frac{\alpha_1}{(\beta_1)^{x_j}}, \beta_1\right) .$$

and during subsequent *re-encryption phase* a block of the form:

$$\left(\frac{\alpha_0}{(\beta_0)^{x_j}} \left(\frac{\alpha_1}{(\beta_1)^{x_j}}\right)^{k_1}, \beta_0(\beta_1)^{k_1}; \left(\frac{\alpha_1}{(\beta_1)^{x_j}}\right)^{k_2}, (\beta_1)^{k_2}\right)$$

is computed for some random k_1, k_2. After decryption phase, J_1 can read the next destination from one of the blocks. In order to hide how far the onion is from its destination, the fully decrypted block is not removed (shortening the package), but instead they are replaced by random contents. Then all blocks are shuffled (randomly permuted or sorted, etc.). Notice that the encoding scheme ensures that two onions encapsulating the same message will look completely different. Hence, the scheme is robust against repetitive attack.

2.5. Navigators

Let us note that the URE-ciphertext of 1 can be treated as some kind of container into which message m can be inserted, by multiplying the first element of quadruple

$$(\alpha_0, \beta_0; \alpha_1, \beta_1) = \left(y^{k_0}, g^{k_0}; y^{k_1}, g^{k_1}\right)$$

by m. So we can “inject" m into such “empty" ciphertext, and this yields a valid URE-ciphertext of m. A similar situation occurs in case of the URE-onions. If some message block encodes 1, one can insert any message into it by simply multiplying the first component of the block by m. Since the URE-onion is used as a “container” encoding some anonymous path in this context, it will be called *navigator* (in [15] more general forms of navigators are used). Navigators have many applications. For instance, they can be used as a kind of anonymous return channel. Namely, a node receiving a navigator which encodes some path to the sender, can insert its own message into the navigator and send it as an URE-onion.

3. Hiding Data Sources

In this section we present mechanisms for hiding identities of P2P nodes holding certain data. The protocol combines two seemingly contradictory features: accessibility of data and anonymity of the party holding it. The main idea behind our protocol is that instead of direct requests, data x can now be fetched only by sending a request to one of the *access points* for x. These access points do not store x, but instead are able to contact the node holding x via some *access path* consisting of λ intermediate nodes. As a result, access points do not learn identity of the node storing x, and the data sources remainhidden from network participants. In addition to this, we introduce a mechanism allowing dynamic changes of the access paths. We show that this modification decreases adversary’s chance to detect the node holding x.

3.1. Access paths

Access points For data with identifier x we define access points – the nodes where requests for x can be sent. The access points do not store data x, but only know how to forward a request for x which will eventually reach the node holding this information. Since they do not store values, their number can be relatively large, which is important to achieve protection against adversaries trying to block access to particular contents.

Network addresses of k access points for x are derived from values of $H(x, i)$ for $i = 1, \ldots, k$, where H is some cryptographic hash function. Since function H is public, every network user can find each access point for x.

Access Structure Let us consider data x, node P holding x, and respective access points $A_1, \ldots, A_k$ of x. For each A_i there is an *access path* between A_i and P, consisting of λ intermediate nodes $A_{i,j}$ for $1 \le j \le \lambda$ (see Figure 3.1). Each $A_{i,j}$ keeps a secret key $d_{i,j}$ and a navigator designed to communicate anonymously with the access point A_i. The access point has also a navigator for communication with P.

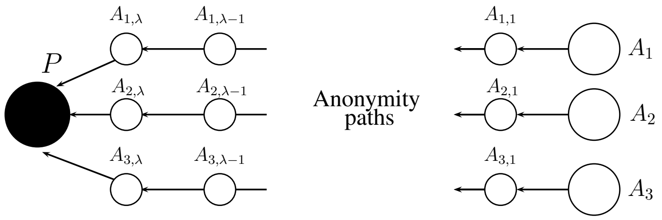


Figure 1. Access structure

Access Initialization In order to initialize access points for x:

- Node P chooses k random paths leading to access points $A_1, \dots, A_k$. For access point A_i, let the access path be $A_i = A_{i,0}, A_{i,1} \dots, A_{i,\lambda}, A_{i,\lambda+1} = P$. For each $0 < j \leq \lambda$ a private key $d_{i,j}$ is generated for $A_{i,j}$; let $y_{i,j} = g^{d_{i,j}}$ be the corresponding public key. The connection between $A_{i,j}$ and $A_{i,j+1}$ is labeled by some unique random identifier $r_{i,j}$ also generated by P.
- For each path $A_{i,1} \dots, A_{i,\lambda}$, node P creates a *raw navigator* N_i using the public keys $y_{i,j}$ for $i = 1, \dots, \lambda$. This navigator has a special form – the block which is to be read by $A_{i,j}$ is computed as:

  $$(r_{i,j} \cdot (y_{i,1} \cdot \dots \cdot y_{i,j})^{k_{i,j}}, g^{k_{i,j}}; (y_{i,1} \cdot \dots \cdot y_{i,j})^{k'_{i,j}}, g^{k'_{i,j}}) ,$$

  where $k_{i,j}$ and $k'_{i,j}$ are random numbers stored by P. This navigator will further get modified (according to path changes) but always allows routing towards P. The message block has the following form:

  $$((y_{i,1} \cdot \dots \cdot y_{i,\lambda})^{k_{i,\lambda}}, g^{k_{i,\lambda}}; (y_{i,1} \cdot \dots \cdot y_{i,\lambda})^{k'_{i\lambda}}, g^{k'_{i,\lambda}}) .$$

  P also prepares an *initialization onion* IN_i. It is a regular onion as described in Section 2.1. It contains an identifier of $A_{i,j+1}$ as well as $r_{i,j}$ in the layer that will be decoded by $A_{i,j}$.
- P sends N_i and IN_i to A_i via an anonymous channel (any protocol of this kind may be used).
- A_i sends IN_i to $A_{i,1}$. Node $A_{i,1}$ peels off IN_i and reads connection identifier $r_{i,1}$ and label of the next server $A_{i,2}$. It stores the pair $(r_{i,1}, A_{i,2})$ for the later use. Then it sends the sub-onion obtained from IN_i to $A_{i,2}$. Nodes $A_{i,2}, \dots, A_{i,\lambda}$ behave in the same way, which eventually creates a connection from A_i to P: each node $A_{i,j}$ learns its own pair $(r_{i,j}, A_{i,j+1})$ which determines the successor on the path to P. For technical reasons (due to the update procedure) $A_{i,j}$ informs $A_{i,j+1}$ via some secure channel about the identifier $r_{i,j}$; node $A_{i,j+1}$ stores a pair $(r_{i,j}, A_{i,j})$ as information on the connection with $A_{i,j}$.

Since initialization onion IN_i is used only once, its encoding need not to be protected against repetitive attack.

Requesting data A user U who wants to get data x sends a request to one of the access points of x, say A_i. After receiving the request, A_i forwards the request through the anonymous path to P. After reaching the request from U, server P sends x to U through some anonymous channel.

Processing a Request A request for x arriving at access point A_i is processed as follows:

Forwarding the request to P: access point A_i receiving a request for x uses navigator N_i and embeds the request into it. Before it is sent by A_i, all fields of the navigator are re-encrypted with some random parameters. Then the navigator is sent to $A_{i,1}$. After decryption phase one of navigator blocks should contain the identifier $r_{i,1}$, which indicates the next node on the path ($A_{i,2}$ for $A_{i,1}$). The navigator N_i is then re-encrypted and sent to $A_{i,2}$. The procedure is repeated until the navigator reaches node P, which decodes the ciphertext containing the request from U.

Delivering x: When P receives a request from U it creates some anonymous path from P to the user node and sends x through this path.

3.2. Paths evolution

Unfortunately, the above scenario does not guarantee anonymity of data sources in case of traffic analysis. It is so, because an adversary may tap nodes (starting with $A_{i,0}$) and listen to the communication. Based on traffic characteristics he may detect identity of subsequent path nodes – $A_{i,1}$, $A_{i,2}$, $\ldots$, and finally reach $A_{i,\lambda} = P$. If no countermeasures are implemented, then it is only the matter of time when the adversary can detect P.

For the reasons mentioned above we implement evolution of access path. We enhance the basic protocol with periodical changes of access paths which are triggered locally. Thus replacement nodes remain unknown to all network nodes except the nodes involved in the change. In this way we make tracing an access path harder. The idea is that a node may leave the path before the adversary observing this node has enough information in order to find the next hop on the path. Should this happen, then the adversary has to move back to the closest node which is still on the path. Further implications of path evolution are discussed in Section 4.

Node replacement During every period of time (of a fixed length) each path node starts a procedure of leaving the path with probability β. We assume that the decision to leave the path is independent from other path nodes. We also assume that the procedures of leaving the path are not executed exactly at the same time at neighbor nodes - standard means for avoiding such a situation may be used. During the replacement procedure other path nodes learn about the replacement itself, but they never learn the identity of a new path member.

Let us assume that a node $A_{i,j}$, $0 < i \leq \lambda$, decides to leave the path. Then the followings steps are executed:

- $A_{i,j}$ chooses at random a node A' that will take over duties of $A_{i,j}$ on the access path. The connections $(A_{i,j-1}, A_{i,j})$ and $(A_{i,j}, A_{i,j+1})$ are to be replaced by connections $(A_{i,j-1}, A')$ and $(A', A_{i,j+1})$, respectively. During the replacement, identifiers $r_{i,j-1}$ and $r_{i,j}$ need to be updated in order to correspond to the new connections.

- $A_{i,j}$ informs A' about the key $d_{i,j}$. Some random key offset δ is chosen locally by A'. Then key $d_{i,j}$ is replaced by $d' = d_{i,j} + \delta$ by node A'. The update $y' = g^{\delta}$ of the public key is shown to $A_{i,j}$.
- The problem now is that the changes of the public key must be reflected in the navigator N_i. For this purpose, $A_{i,j}$ sends y' together with identifier $r_{i,j}$ to P. Of course, $A_{i,j}$ does not know P. However, it can open an anonymous channel to P using onions with requests for x processed by $A_{i,j}$ towards P. Namely, assume that a request is sent by $A_{i,j}$ using an onion N. Right afterwards $A_{i,j}$ may re-send re-encrypted N with the first message block element multiplied by some fixed number π. Then it re-sends N twice – after re-encryption and multiplying the same element by y' and $r_{i,j}$, respectively.
- If P receives some request z and then $z \cdot \pi$, then it means that some $A_{i,j}$ wants to open an anonymous channel. P expects next messages to contain the key update. One of the following messages should therefore contain $z \cdot y'$ and another one $z \cdot r_{i,j}$. So P can retrieve y' and $r_{i,j}$. From the value of $r_{i,j}$ node P recognizes which node on the path has transferred its duties and knows that the public key update of the jth node is y'.
- P informs A_i about necessary changes in N_i. Namely, through some anonymous secure channel it sends all necessary data to A_i: the number j, and for each block of N_i of the form

  $$(u \cdot (y_{i,1} \cdot \ldots \cdot y_{i,t})^{k_{i,t}}, g^{k_{i,t}}; (y_{i,1} \cdot \ldots \cdot y_{i,t})^{k'_{i,t}}, g^{k'_{i,t}}) \ ,$$

  for $t \geq j$, node P sends

  $$(y')^{k_{i,t}} \quad \text{and} \quad (y')^{k'_{i,t}} \ .$$

  Node A_i multiplies the first and third component of this block by the numbers obtained.

Obviously, the same mechanism may be applied in order to update identifiers $r_{i,j-1}$ and $r_{i,j}$. We skip the details here.

Note that in the above scenario node $A_{i,j}$ does not have to prove to P its right to introduce path changes. Obviously, node P does not necessarily trust that this operation is performed by a valid node. Hence, an additional mechanism for node validation is required in order to prevent malicious replacements. Simple, yet effective method to perform validation is to require $A_{i,j}$ to send two additional messages to P. Namely, a message containing the identity of the replacement node A' with a digital signature of $A_{i,j}$.Then P can verify whether replacements are introduced by proper nodes. Now it also has up-to-date information about the access paths. Note that this does not harm security of our scheme, since our main concern is to hide data source, and not the intermediate nodes from the data source.

4. Resistance to Dynamic Adversary

In order to protect against adversaries trying to track down data sources we have introduced access paths evolution mechanism, where decisions how to change are performed independently and randomly by the nodes on the paths – with no central control. This should provide high protection against adversaries who tap intermediate nodes and perform traffic analysis. Let $S_0, S_1, \ldots, S_\lambda$ denote an access path from the access point S_0 to the node $P = S_\lambda$. The adversary will tap S_i (starting with $i = 0$) and analyze incoming and outgoing traffic trying to determine S_{i+1}. For simplicity we assume that during each round the adversary may discover identity of S_{i+1} with probability α independently of the attack history. The procedure is repeated until the adversary reaches P. We assume that the adversary makes no mistakes – either he guesses S_{i+1} or remains observing S_i. Obviously, without path updates the adversary would reach P in expected time λ/α.

Path Evolution and the Adversary If there was no evolution of the paths, an adversary who reached some server on the path, would know for sure all nodes between this server and the access point starting the path. Since path evolution takes place, nodes previously known to the adversary are supposed to leave the path randomly. As a result, only some subset of the path prefix remains known to the adversary. For simplicity of the analysis we assume that once the adversary guesses that some node is on a path, afterwards he can always verify whether it still belongs to this path.

The first observation is that the further the adversary advances (say up to S_i), it becomes more likely that the nodes left behind have already left the path. The probability that a given node has left the path grows with its distance to S_i. Additionally, there is also a fair chance that the adversary will "choke" at some point of the path – this happens when the node tapped leaves the path! In such case the adversary has to backtrack to the closest known path node – the one that is known to the adversary and has not left the path in the meantime. If there is no such node left, the attack must start from the very beginning.

The attack can be described by the following stochastic process:

- the adversary performs a random walk on a path of length λ, he starts from the left and the goal is to reach the opposite side of the path,
- at each step with probability α the adversary may move one step right,
- the nodes visited by the adversary may be *marked* – a node becomes marked, when the adversary enters this node from the left,
- during each step each marked node becomes unmarked with probability β; unmarking occurs independently for each node,
- if the node currently pointed by the adversary becomes unmarked, he has to backtrack to the rightmost marked node.

This process corresponds to the situation that the adversary monitors only the last known node on the path. We should also consider another type of adversary, called a *strong adversary*, who monitors all known nodes on path prefix and is always trying to fill path "gaps" (unknown intermediate nodes). In case of a strong adversary:

- if a node is unmarked and its predecessor is marked, then it becomes marked with probability α;

The crucial question is how fast can the adversary advance to the right. Of course, the answer depends on α and β. Generally we may assume that $\alpha \leq \beta$, since performing traffic analysis is always time consuming, while exchanging nodes on the path can be performed fast and with little effort. The main problem is how large must be β compared to α so that the adversary would have to perform a very large number of steps in order to reach the end of the access path.

The most important point for the adversary is to keep moving forwards, because if he stays longer at some position, then more and more nodes behind him get unmarked. As a result, backtracking may require a large number of positions. Additionally, in a long period of time there is fair chance that a short series of failures appear (un-markings of the current node). When this happens and the number of marked nodes behind the adversary is smaller than the length of this series, the adversary has to start from the beginning. In the following subsections we shall see that such a situation occurs quite frequently.

4.1. Probability of Adversary's Success - Analytical Approach

4.1.1. Number of Nodes behind the adversary

The adversary taps S_l for some $l < \lambda$. Our aim is to estimate the number of nodes behind S_l that remain marked. Let X_i be a random variable indicating that node S_i is still marked. Then $Y_j = \sum_{i=1}^{j} X_i$ is the number of marked nodes up to S_j. Let us investigate the distribution of X_i. Of course it depends on the number of steps that have passed after marking this node by the adversary:

$$\Pr(X_i = 1) < (1-\beta)^{l-i} \, .$$

Indeed, at least $l - i$ rounds has passed after the last visit of the adversary in S_i - he needs at least $l-i$ moves to reach S_l. In fact, the actual values are usually much smaller, since the advancement of the adversary is significantly slower. It is clear that $\mathbb{E}[X_i] = \Pr(X_i = 1)$. So by linearity of expectation,

$$\mathbb{E}[Y_{l-1}] < \sum_{i=1}^{l-1}(1-\beta)^{l-i} = (1-\beta)\frac{1-(1-\beta)^{l-1}}{\beta} \, .$$

Now let us estimate the variance of Y_{l-1}. We have $\mathbb{E}[X_i] = \mathbb{E}[X_i^2]$, since random variables X_i^2 and X_i have exactly the same distribution. Since $\mathbb{E}[Y_j] > 0$, we have $\mathbb{V}ar[Y_{l-1}] = \mathbb{E}[Y_{l-1}^2 - (\mathbb{E}[Y_{l-1}])^2] < \mathbb{E}[Y_{l-1}^2]$. Now

$$\mathbb{E}[Y_{l-1}^2] < \mathbb{E}[(X_1 + \ldots + X_{l-1})^2] = \sum_{i=1}^{l-1}\mathbb{E}[X_i^2] + \sum_{i \neq j}\mathbb{E}[X_i X_j] \, .$$

Since X_i and X_j are independent random variables we have

$$\Pr(X_i X_j = 1) < (1-\beta)^{2l-i-j}$$

and so

$$\begin{aligned}
\mathbb{E}[Y_{l-1}^2] &< \mathbb{E}[Y_{l-1}] + \textstyle\sum_{i\neq j}(1-\beta)^{2l-j-i} \\
&< (1-\beta)\tfrac{1-(1-\beta)^{l-1}}{\beta} + \Big(\big(\textstyle\sum_{i=1}^{l-1}(1-\beta)^{l-i}\big)^2 - \textstyle\sum_{i=1}^{l-1}((1-\beta)^{l-i})^2\Big) \\
&= (1-\beta)\tfrac{1-(1-\beta)^{l-1}}{\beta} + \Big((1-\beta)\tfrac{1-(1-\beta)^{l-1}}{\beta}\Big)^2 - \tfrac{(1-\beta)^2(1-(1-\beta)^{2l-2})}{1-(1-\beta)^2} \, .
\end{aligned}$$

Now, using the approximation of variance and expected value of Y_{l-1} one can see that the chance that many nodes except the current node S_l of the adversary are still marked is indeed slim. For this purpose we can use Tchebychev's inequality. For instance, for $\beta = 1/2$ the chance that two or more previously visited nodes are still marked is smaller than $1/30$. Note, that this approximation does not depend on actual position on the path (parameter l) - the estimation holds for any position, even those close to the end of the path.

4.1.2. Probability of two Consecutive Successes during k Bernoulli Trails

Let us consider a sequence of k Bernoulli trials, where probability of success in each trial is γ. Then the probability $\Pr\{\gamma, k\}$ that at least once during the sequence we achieve two successes in a row for $k \geq 2$ can be computed as $1 - p_1(k) + p_2(k)$ where

$$p_1(k) = \frac{\left(\frac{1-\gamma-\sqrt{\Delta}}{2} - (1-\gamma^2)\right)\left(\frac{1-\gamma+\sqrt{\Delta}}{2}\right)^{k-1}}{-\sqrt{\Delta}}$$

$$p_2(k) = \frac{\left((1-\gamma^2) - \frac{1-\gamma+\sqrt{\Delta}}{2}\right)\left(\frac{1-\gamma-\sqrt{\Delta}}{2}\right)^{k-1}}{\sqrt{\Delta}}$$

and

$$\Delta = \sqrt{(1-\gamma)^2 + 4(1-\gamma)\gamma}\,.$$

In particular for $\gamma = 1/2$ we get:

$$\Pr\{1/2, k\} = 1 - \left(\frac{5+3\sqrt{5}}{10}\right)\left(\frac{1+\sqrt{5}}{4}\right)^k - \left(\frac{5-3\sqrt{5}}{10}\right)\left(\frac{1-\sqrt{5}}{4}\right)^k.$$

Using this formula we can estimate that in series of $k = 10$ trials two consecutive successes will occur with probability $\Pr\{1/2, k\} > 0.75$. For $k = 20$ trials, we have $\Pr\{1/2, k\} > 0.95$. One can easily see that if the adversary has to backtrack in two consecutive rounds, and no more than one marked node remains on the path, he will return to the access point and will have to start the attack from the beginning. Unfortunately for the attacker, probability of such an event is pretty high.

4.2. Experimental results

Although the analytical estimations presented in previous subsection are based on pessimistic assumptions, the phenomena discovered are confirmed by our experimental results. The following table presents results of experiments consisting of 100.000 trials performed for each choice of parameters. The probability that a given node is replaced by another node within a single step equaled $\frac{1}{2}$, and the probability that an adversary finds the next path node during a single step was $\frac{1}{K}$.

experiment 1:	0 0 -1
experiment 2:	1 1 1 0 0 1 0 -5
experiment 3:	1 1 1 1 1 0 -1 1 0 1 0 -7
experiment 4:	0 0 0 -1
experiment 5:	0 0 -1
experiment 6:	1 1 1 1 0 -5
experiment 7:	0 1 0 1 0 0 0 0 0 -3
experiment 8:	1 1 1 0 -1 1 0 1 1 0 1 1 0 1 1 0 -2 0 0 -8
experiment 9:	0 1 0 -1 0 0 0 1 1 0 -2 1 0 -1 1 0 1 1 0 1 0 0 -5
experiment 10:	0 0 0 0 0 0 0 0 -1
experiment 11:	1 1 0 -1 0 0 0 1 1 0 1 0 -1 0 -4
experiment 12:	0 0 1 1 1 1 0 -5
experiment 13:	0 1 1 1 1 0 1 1 0 -1 1 0 0 -7
experiment 14:	1 1 1 1 0 -5
experiment 15:	1 0 -1 1 0 1 0 0 -3

Table 2. trajectories of the adversary

	path length = 10			path length = 15			path length = 20		
K	2	3	4	2	3	4	2	3	4
weak adversary									
50 steps	23803	1806	230	3631	63	3	556	2	0
100 steps	45627	4271	531	9204	147	7	1594	4	1
strong adversary									
50 steps	50651	6403	712	17787	480	15	4273	19	0
100 steps	82442	16426	2036	47645	2193	62	22872	228	2

Table 1. the number of successful attacks

The following phenomena can be observed:

- there is a big difference between a strong adversary and a weak adversary that monitors only the last known node. However, increasing frequency of path changes compensates for this problem,
- if path length is 20 and the rate of path change is 2 times bigger than the advancing adversary, then he succeeded for none of 100.000 trials to reach the end of the path within 50 steps– regardless of the adversary model.

Let us illustrate some typical "trajectories" of the adversary: the numbers illustrate how many steps forward are done during consecutive steps (the leftmost number denote the advance during the first step). Backtracking i positions is marked as $-i$. A trajectory terminates, once the adversary returns to the access point. This occurs if the current position of the adversary gets unmarked and there are no other marked positions left.

These examples show that after traveling along the path for a longer time it happens very often that the adversary is returned to the beginning of the path in just one step.

4.3. *Attack success chances – analytical approach*

Determining probability of reaching the right end of the path, when various protocol parameters are used, helps to understand adversary's chances. We have determined exact values of success probabilities, as a function of α and number of attack rounds. The computation was performed for paths of length 8 with $\beta = 0.5$ and $\alpha = 0.05, 0.10, \ldots, 0.45, 0.50$. Figure 2 presents the results. Numerical values are given

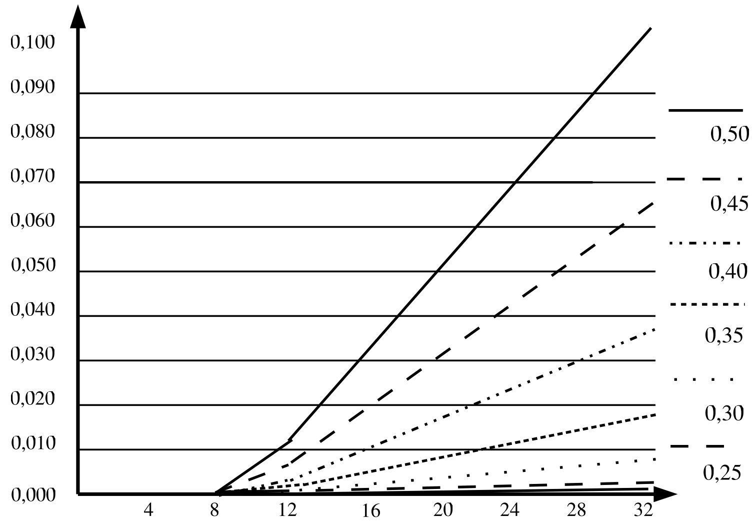


Figure 2. Probabilities of attack success

in the following table. The columns represent respective α's and rows are computed for different number of attack rounds (20, 24, 28 and 32).

	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50
20	0.000	0.000	0.000	0.000	0.001	0.003	0.008	0.017	0.031	0.051
24	0.000	0.000	0.000	0.000	0.002	0.005	0.012	0.024	0.043	0.069
28	0.000	0.000	0.000	0.001	0.002	0.006	0.015	0.031	0.054	0.085
32	0.000	0.000	0.000	0.001	0.003	0.008	0.019	0.037	0.065	0.102

Table 3. Probabilities of attack success - numerical values

As can be seen, even for short paths of length 8 an adversary needs many rounds to raise the chance of reaching path end up to 0.1. This happens even for $\alpha = \beta = 0.5$. Should α fall below this value, the chances drop below 0.01 just for $\alpha = 0.3$. What can also be observed is that for $\alpha = 0.5 \cdot \beta$ the probability of success drops to 0.003 which is 30 times smaller than in case of $\alpha = 0.5 \cdot \beta$. This leads to the following conclusions:

- even for moderate access path lengths the adversary is expected to need many rounds,
- the protocol can be refined when the ratio α/β can be estimated.

5. Final Remarks

The process of gaining knowledge about the access paths by the adversary can be modeled stochastically and analyzed. Certainly, our approach in this paper is not the only possible one. Our goal was just to support the protocol design and show its features, rather than fully explore the stochastic behavior of the attacks. Further research on this topic is possible.

References

[1] Chu Yee Liau, S. Bressan, Kian-Lee Tan, Efficient Certificate Revocation: A P2P Approach, *Workshop on Southeast Asian Computing Research (ASIAN)*, 2002.

[2] Lidong Zhou, F.B. Schneider, R. van Renesse, COCA: A Secure Distributed Online Certification Authority, *ACM Transactions on Computer Systems* **20(4)** (2002) 329-368.

[3] Aameek Singh, Ling Liu, TrustMe: Anonymous Management of Trust Relationships in Decentralized P2P Systems, *3rd IEEE International Conference on P2P Computing*, Linköping, 2003, 142-149.

[4] *TURTLE. P2P Architecture for Safe Sharing of Sensitive Data*, project Web page: `http://turtle-p2p.sourceforge.net/`

[5] M.K. Reiter, A.D. Rubin, Crowds: Anonymity for Web Transactions, *ACM Transactions on Information and System Security* **1(1)** (1998) 66-92.

[6] M.J. Freedman, R. Morris, Tarzan: A peer-to-peer Anonymizing Network Layer, *9th ACM Conf. on Computer and Communications Security*, 2002, 193-206.

[7] D.M. Goldschlag, M.G. Reed, P.F. Syverson, Hiding Routing Information, Information Hiding '1996, *Lecture Notes in Computer Science* **1174**, Springer-Verlag, 137-150.

[8] C. Gülcü, G. Tsudik, Mixing E-mail with BABEL, *ISOC Symposium on Network and Distributed System Security*, IEEE, 1996, 2-16.

[9] R. Dingledine, N. Mathewson, P. Syverson, Tor: the Second Generation Onion Router, *USENIX Security*, 2004.

[10] C. Rackoff, D.R. Simon, Cryptographic Defense Against Traffic Analysis, *25th ACM Symposium on Theory of Computing*, ACM, 1993, 672-681.

[11] R. Berman, A. Fiat, A. Ta-Shma, Provable Unlinkability Against Traffic Analysis, Financial Cryptography (FC)'2004, *Lecture Notes in Computer Science* **3110**, Springer-Verlag, 266-280.

[12] M. Gomułkiewicz, M. Klonowski, M. Kutyłowski, Provable Unlinkability Against Traffic Analysis already after $\mathcal{O}(\log(n))$ Steps!, Information Security Conference (ISC)'2004, *Lecture Notes in Computer Science* **3225**, Springer-Verlag, 354-366.

[13] P. Golle, M. Jakobsson, A. Juels , P. Syverson, Universal Re-encryption for Mixnets, *The Cryptographers' Track at the RSA Conference*'2004, *Lecture Notes in Computer Science* **2964**, Springer-Verlag, 163-178.

[14] M. Gomułkiewicz, M. Klonowski, M. Kutyłowski, Onion Routing Based On Universal Re–Encryption Immune Against Repetitive Attack, Workshop on Information Security Applications (WISA) '2004, *Lecture Notes in Computer Science* **3325**, Springer-Verlag, 400-410.

[15] M. Klonowski, M. Kutyłowski, F. Zagórski, Anonymous Communication with On-line and Off-line Onion Encoding, SOFSEM'2005, *Lecture Notes in Computer Science* **3381**, Springer-Verlag, 229-238.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Efficient Broadcast from Trapdoor Functions

Yi MU [a,1], Willy SUSILO [a] and Xinyi HUANG [b]
[a] *University of Wollongong*
[b] *Nanjing Normal University*

Abstract. We present a novel scheme of broadcast encryption that is suitable for broadcast servers such as pay TV services. The important feature of our scheme is that the length of a broadcast string in our scheme is independent of the number of receivers in the system; hence it is suitable for large groups. Our scheme is based on a trapdoor encryption technique under the RSA assumption. We also describe a variant of our scheme which provides stronger security.

Keywords. Broadcast encryption

1. Introduction

Pay TV broadcasting schemes can be related to broadcast encryption, which allows a sender to deliver information to a group of users; each holds a different decryption key. The broadcast encryption was introduced by Fiat and Naor [7]. Since then, there have been a number of schemes in the literature (e.g., [12,8,9,11]). Those schemes vary from bounded to unbounded number of broadcasts. They may be composed of either fixed user groups or variable (or dynamic) user groups. Most broadcast encryption schemes allow a server to deliver information to a set of users that can be dynamically formed. Namely, the broadcaster can determine which users will receive the information with the pre-defined user information. Each authorized user can recover the information by using the corresponding secret key.

A typical Pay TV system consists of a broadcaster and a number of subscribers. The broadcaster broadcasts TV programs to its subscribers. When a Pay TV program is transmitted through an optical fibre or a microwave network, the protection of the program must be enforced against non-subscribers and also the subscribers who want to forge new decryption keys. A pay TV scheme can be achieved with a symmetric-key scheme, where all receivers share the same decryption key. Although it has the advantage of computational efficiency, the key management is often problematic. Public-key schemes in pay TV allow each receiver to hold a different decryption key. Therefore, revocation can be easily done by the broadcaster.

There is a tradeoff between the following two scenarios in broadcast encryption: perfect user revocation and ideal computational efficiency.

[1]Correspondence to: Yi Mu, School of IT and Computer Science, University of Wollongong, Wollongong, NSW 2522, Australia. Tel.: +61 2 4221 5228; Fax: +61 2 4221 4170; E-mail: ymu@uow.edu.au.

To realize Scenario 1, we must assume that the encryption key is dependent on the number of receivers and the size of encryption data is proportional to the number of users. Although the broadcaster can easily add or remove a user, the drawback is obvious: the encryption key and the associated data must be changed whenever a user is added or removed. There are various realizations of Scenario 1 (eg. [4,7,9]). The recent invention of identity-based cryptography has also made identity-based broadcast (broadcast based on identities of users) becomes feasible [3].

Scenario 2 gives us a much more efficient way in handling broadcast encryption, since the encryption key and associated parameters can be kept the same and independent of the number of receivers. This feature even stands when a new user is added to the system. The drawback is due to revocation. It is hard to remove a user from the system. The only realization of the scenario is due to Narayanan *et al.*[13]. They also tried to sort the revocation problem out by introducing a new parameter for each program; namely, the user receives such a parameter provided he subscribes the specific program. It does not satisfy the ultimate goal of user revocation; that is, any user should be able to be removed from the system by the broadcaster whenever he wants to.

Therefore, there is a tradeoff between Scenario 1 and Scenario 2. If we want to achieve Scenario 1, we have to make a compromise due to computational overhead. On contrast, if we want to achieve Scenario 2, then we will not have a revocation advantage. In this paper, we are not going to find a solution to this tradeoff in which we believe there exists no any desirable solution. Instead, we will propose a new realization of Scenario 2 with a more efficient encryption algorithm and provide a simple and effective revocation scheme that accommodates the basic needs in pay TV broadcast. We also give a variant of our scheme which is secure against IND-CCA2 attacks.

Bellare *et al.*[2] pointed out the relation between many-to-one trapdoor functions and public-key cryptosystems. They found that many-to-one trapdoor functions can be constructed from public-key cryptosystems. Our interest is different. We are interested in how to construct a public-key scheme from a many-to-one trapdoor function. Our new scheme is based on the trapdoor algorithm that has been widely studied in the literature [5,6,10]. We take advantage of the algorithm to construct the cryptographic keys such that one encryption key maps multiple decryption keys. The security of our scheme is based on the RSA assumption.

The rest of this paper is arranged as follows. In section 2, we will give a set of definitions associated with our schemes and security consideration. Section 3 describes our new scheme based on the trapdoor technique and the security proofs to our scheme. Section 4 presents is a variant of our scheme, which is secure against IND-CCA2 attacks. The final section is our conclusion.

2. Definitions

In this section, we describe the formal definitions of our scheme and give the security definition.

Definition 1 *Our broadcast scheme consists of the following four phases:*

- Setup*: A probabilistic algorithm that on input a security parameter ℓ, outputs definitions of the set of users $\mathcal{U}$, the broadcaster B, the message space $\mathcal{M}$, and the* *ciphertext space $\mathcal{C}$. Each user in $\mathcal{U}$ obtains the associated private key k_{ij} corresponding to the encryption keys y_j. For the system with n users and m programs, we have $1 \leq i \leq n$ and $1 \leq j \leq m$. All other parameters are denoted by π. For revocation purposes, we assume that each user also obtains a symmetric key κ_i shared with the broadcaster.*

- Subscribe*: The user i subscribes program j from the broadcaster and obtains a decryption key k_{ij}, encrypted with κ_j.*
- Encrypt*: a probabilistic algorithm that on input (M, y_j), where $M \in \mathcal{M}$ and y_j is the encryption key, outputs a ciphertext tuple (C, ρ), where $C \in \mathcal{C}$ and ρ denotes the remaining parameters.*
- Decrypt*: A deterministic algorithm that on input (C, ρ) and a valid decryption key, outputs the message M.*

Based on the definition above, we will give two concrete schemes. Our schemes are mixed with ElGamal encryption and the RSA assumption, but not incorporating the standard setting for ElGamal encryption. Tsiounis and Yung have given a general study of ElGamal encryption security [15]. They showed that ElGamal encryption scheme is as secure as the Decisional Deffie-Hellman problem. Our scheme differs from this, its security is based on the RSA assumption. We define security in terms of the sense of indistinguishability. Intuitively, if it is infeasible for an adversarial algorithm to distinguish between the encryption of any two messages, even if these messages are given, then the encryption is secure. Our first scheme is not secure against IND-CCA2 [**?**], but our second scheme is. We allows the attacker to access the decryption oracle even after he has received the challenge (the ciphertext).

Definition 2 *(Security of the first scheme) Let (*Setup, Encrypt, Decrypt*) be an encryption scheme. If we say it is secure in the sense of indistinguishability and intractability of the RSA assumption, then there exists no polynomial-time adversarial oracle $\mathcal{A}$ that, on input a ciphertext, outputs the original message.*

Definition 3 *(Security of the second scheme) Let (*Setup, Encrypt, Decrypt*) be an encryption scheme. If we say it is secure against IND-CCA2, then there exists no polynomial-time adversarial oracle $\mathcal{A}$ that can solve the RSA problem in polynomial time and on input a ciphertext, outputs the original message.*

Definition 4 *(Collusion Resistant) Given η decryption keys $\{k_i\}$ for $1 \leq i \leq n'$, there exist no polynomial forgers who can collaboratively find a valid decryption key χ such that $\chi \notin \{k_i\}_{i=1,\cdots,n'}$.*

We consider the scenario that the forged key is not necessarily a key generated by the broadcaster previously. It can be any form as soon as it can be used to decrypt a ciphertext generated by the broadcaster.

2.1. Trapdoor Based on the RSA Assumption

In this section, we describe the trapdoor construction, which has been studied in the literature [5,6,10]. We will utilize this trapdoor construction to our new broadcast scheme in the next section.

We consider the RSA setting. Let N is the composition of two safe primes p and q. Set $\phi(N) = (p-1)(q-1)$. Select an integer $e \in \mathbb{Z}^*_{\phi(N)}$ relatively prime to $\phi(N)$. The trapdoor is defined as follows.

Definition 5 *(Trapdoor) Given $g \in \mathbb{Z}^*_N$ of order $\phi(N)$, there exist a set of integers (a_i, b_i), for $i = 1, \cdots, n$, such that $y = g^{a_i} b_i^e \bmod N$ is a constant, where a_i are selected from $\mathbb{Z}_{\phi(N)}$ and b_i are selected from $\mathbb{Z}^*_N$.*

The number of trapdoors that can be found are dependent on the value of $\phi(N)$. It is trivial to find suitable (a, b) for $a \in \mathbb{Z}_{\phi(N)}$ and $b \in \mathbb{Z}^*_N$ that for given $g, y \in \mathbb{Z}^*_N$ and e be a prime chosen from $\mathbb{Z}_{\phi(N)}$, $y = g^a b^e \bmod N$ forms a trapdoor. We find that given a value $y = g^a b^e \bmod N$ along with g and $1/e$, for each value $a' \neq a$, it is trivial to find a unique value b' such that $y = g^{a'} b'^e \bmod N$. Note that $b' = (yg^{-a'})^{1/e} \bmod N$.

For convenience in the presentation, we will omit modulus if it is clear.

Theorem 1 *If (c, N, g, e) defined above are given, the trapdoor value is $x = g^{1/e}$. [10]*

Proof (Sketch): We prove that given x along with g, e, a trapdoor can be constructed. Given values of (x, a, b), compute $d = x^a b$. The trapdoor can be formed by raising the e-roots on both sides. We find $y = d^e = g^a b^e$. We can find another pair (a', b') by randomly selecting a value a' and computing $b' = dx^{-a'}$. Obviously, $y = g^{a'} b'^e$. □

If e is a fixed public value, then the security of the trapdoor is based on the RSA assumption:

Definition 6 *(RSA assumption) Let N is the composition of two safe primes p and q. Set $\phi(N) = (p-1)(q-1)$. Let e be an integer relatively prime to $\phi(N)$. Given a random element s selected from $\mathbb{Z}^*_N$ and a fixed $e \in \mathbb{Z}_{\phi(N)}$, it is hard to find x such that $x^e = s \bmod N$.*

We will see that this assumption is sufficient to our scheme, since we do not require e to vary. Most previous applications of this kind of trapdoors are based on the strong RSA assumption where e can be chosen by the attacker.

3. The Basic Scheme (BS)

In this section, we describe our new scheme based on the trapdoor discussed above. The basic idea for our construction is to achieve one to many maps by taking advantage of trapdoor.

Before going to the scheme in detail, we briefly describe how the scheme works. There is a broadcast server Bob who broadcasts several programs to the valid subscribers. Any user who wants to get the service must register with Bob first to get a permanent subscription key shared with Bob. Bob possesses a set of broadcast encryption keys, one for each program. The subscription keys are used to deliver the program keys to the users who have subscribed the program, respectively. These keys can be used for revocation, namely, Bob can refuse to send the user further program keys if the user has not paid. For a program, each user holds a different key, all map to the program encryption key.

The scheme is given as follows in terms of Setup, Encrypt, Subscribe, and Decrypt phases as defined previously in this paper.

Setup: The broadcaster sets the system up by selecting two large primes p, q, setting $N = pq$ and $\phi(N) = (p-1)(q-1)$. He also finds a number $e, \theta \in \mathbb{Z}^*_{\phi(N)}$ relatively prime to $\phi(N)$, a public generator, $g \in \mathbb{Z}^*_N$, and sets $\mathcal{M} = \mathcal{C} = \mathbb{Z}^*_N$. Each user obtains a pair of secret key from the broadcaster. For User i, the secret key pair denoted by $k_i = (a_i, \hat{b}_i)$, where $a_i \in \mathbb{Z}_{\phi(N)}$ and $\hat{b}_i = b_i^{\theta e}$. All $\{a_i, \hat{b}_i\}$ maps to a single value of y_j for program j. In other words, for a fixed y_j, we have $y_j = g^{a_i} b_i^e \bmod N$, $i = 1, \cdots, n$. which can be constructed as follows: select $a_i \in \mathbb{Z}_{\phi(N)}$ and then compute $b_i = (y_j g^{-a_i})^{1/e}$. The public parameter is N only. We have assumed that y_j is associated with a single problem in the Pay TV system. For a multi-program system, a suitable y_j is selected for each program.

Subscribe: A user wishes to subscribe a program or programs, he or she should register with the broadcaster and obtains a permanent subscription key κ_i, which is a symmetric key such as an AES key. This key is used to encrypt the corresponding program key which is then sent the subscriber. For example, if user i has subscribed program j, then encrypted $(a_i, \hat{b}_i)$ along with the related parameters are sent to user i.

Encrypt: To broadcast a message M of program j, the broadcaster selects a random $r \in_R \mathbb{Z}_{\phi(N)}$, computes the broadcast triplet $(My_j^r, g^r, r/\theta)$, where $M \in \mathcal{M}$, $g \in \mathbb{Z}^*_N$, and $e \in \mathbb{Z}^*_{\phi(n)}$. Then, the triplet (A, B, C) is broadcasted to all users.

Decrypt: Upon receiving the broadcast triplet (A, B, C), any user who has previously subscribed program j can retrieve M_j by computing $A(B^{a_k} \hat{b}_k^C)^{-1} = M_j$ for $(a_k, \hat{b}_k) \in \{a_i, \hat{b}_i\}_{i=1,\cdots,n}$.

The correctness of the scheme is obvious: all users who hold a valid program decryption key can retrieve the program. However, it has to be sound, i.e., only legitimate subscribers can retrieve the program. We discuss this issue in the next section.

3.1. Security of BS

We consider security in our scheme as two aspects:

- Attacks from outsiders who have not got a valid decryption key and attempts to find a valid decryption key that can be used to decrypt any broadcasted ciphertext in the corresponding program.
- Attacks from insiders, each has got a valid decryption key and attempts to find another valid decryption key by collusion.

For outsider attacks, we consider the security in our scheme in the sense of indistinguishability [15]. We can refer our encryption scheme (omit the Subscribe phase) to as a variation of ElGamal encryption. Following the definition of indistinguishability [15], we have the following definition.

Definition 7 *(Indistinguishability) An encryption scheme (*Setup, Encrypt, Decrypt*) is said to be secure in the sense of indistinguishability, if, for every polynomial time* *algorithm F for every probabilistic polynomial time algorithm $\mathcal{A}$, for every constant $\tau > 0$ and for every sufficiently large ℓ,*

$$\Pr\left[F(1^\ell) = (\alpha, \beta, \gamma)\ s.t.\ \Omega(\alpha, \beta, \gamma) > \frac{1}{\ell^\tau}\right] < \frac{1}{\ell^\tau}.$$

$$\Omega(\alpha, \beta, \gamma) =$$

$$\left|\Pr[\mathcal{A}(\gamma, \mathsf{Encrypt}_{\mathsf{Setup}(1^\ell)}(\alpha) = 1] - \Pr[\mathcal{A}(\gamma, \mathsf{Encrypt}_{\mathsf{Setup}(1^\ell)}(\beta) = 1]\right|.$$

Here, $\alpha, \beta \in \mathcal{M}$ and γ is a polynomial random variable.

3.1.1. Security against Outsiders.

Although our scheme is a variation of the ElGamal encryption scheme, the security of our scheme is not based on the decision Diffie-Hellman problem but the RSA problem.

Let us take a look at why it is not based on the DDH problem. Given a valid ciphertext triplet $(My^r, g^r, r/e)$, the associated DDH triplet should be $(g^r, y = g^\phi, g^\chi)$. That is, given g^r, g^ϕ, decide if $\chi = \phi r$. However, in our scheme, y is not public.

Theorem 2 *If BS is not secure in the sense of indistinguishability, then there exists a probabilistic polynomial time adversary that can solve the RSA problem with overwhelming probability.*

Proof: If our scheme is not secure in the sense of indistinguishability it suffices to show that we can find with non-negligible probability a pair of plaintext messages such that their encryption can be distinguished with non-negligible probability of success.

We first show that given a valid encryption triplet (A, B, C) and the public information N, the security of our scheme can be reduced to a RSA problem. Observe $A = MB^a b^C$. The adversary chooses random a' and computes $A(B^{a'})^{-1}$ which should give the equality $A(B^{a'})^{-1} = b'^C$ if the RSA problem can be solved and b' can be found in polynomial time.

Assume there exists a RSA oracle. The game for the RSA assumption is as follow:

Game 1: Given (A, B, C) and N,

G1-1: Select random $a' > 1$.
G1-2: Compute $s \leftarrow A(B^{a'})^{-1}$.
G1-3: Select random $b' \in \mathbb{Z}_N^*$.
G1-4: Test $b'^C \stackrel{?}{=} s$. If yes, output 1, otherwise 0.

The adversary plays Game 1 and asks q_1 queries to the RSA oracle. The probability of success is $q_1/2^{2\ell}$.

We then show that if the RSA oracle outputs 1, the adversary can distinguish the ciphertext for messages m_0, m_1. Our adversarial algorithm selects random $m_0, m_1 \in \mathbb{Z}_N^*$. Then given the encryption of these messages:

$$(m_i y^{r_0}, g^{r_0}, r_0/\theta) \leftarrow \mathsf{Encrypt}(m_i),$$

$$(m_{1-i} y^{r_1}, g^{r_1}, r_1/\theta) \leftarrow \mathsf{Encrypt}(m_{1-i}),\ i \in_R \{0, 1\},$$

where $r_0, r_1 \in_R \mathbb{Z}_{\phi(N)}$, we only need to show given that the RSA oracle outputs 1, the adversary can distinguish non-negligibly better than random guessing which ciphertext encrypts which message (find i). The success probability of random guess is $\frac{1}{2} + \epsilon$ for a small number ϵ.

If the adversary can solve the RSA problem, then he can output: $(a', \hat{b}')$, $y^{r_0} = B^{a'}\hat{b}'^{C}$, and

$$m_i y^{r_0}/m_0 = \begin{cases} y^{r_0} & (i = 0) \\ m_i y^{r_0}/m_0 & (i = 1) \end{cases}$$

In this instance, the adversary is sure that the first ciphertext encrypts the first message with probability non-negligibly better than random guessing. Of course, if the RSA problem is intractable or the RSA oracle outputs 0, then the adversary cannot determine y^{r_0}. He can only randomly pick i. The advantage of the adversary is $\Pr[\mathsf{win}] - \frac{1}{2}$. □

3.1.2. Security against Insiders.

We consider the scenario that several valid users collude to find a valid decryption key which is not one of keys they currently hold. We will show that a collusion will not give the adversaries only advantage in gaining a new pair of decryption key.

Definition 8 *(Insider Attacks) Given a set of decryption keys $\{k_i\}_{i\in\mathcal{F}} \subseteq \{k_j\}_{j=1,\cdots,n}$, where $\mathcal{F}$ is the set of indices for the set of forgers in $\mathcal{U}_s$ which denotes a set of legal subscribers, find a new decryption key k_f such that $f \notin \mathcal{F}$, where k_f is a valid key that decrypts all messages belonging to the same program.*

Remark: Since r is unique to the ciphertext in the program, a successfully forged decryption key must be independent to r. For example, we consider the following case to be a unsuccessful forgery. Given a valid key $k = (a, \hat{b})$ and a valid ciphertext (A, B, C), we select a' at random and try to find the corresponding $\hat{b}'^{C}$ from $\hat{b}'^{C} = B^{(a-a')}\hat{b}^{C}$. Although $\hat{b}'^{C}$ along with a' can also be used to decrypt (A, B, C), it is dependent on r and cannot be used to decrypt other ciphertexts in the program.

Theorem 3 *Our scheme is secure against collusion attacks from insiders if the RSA problem is intractable.*

Proof (Sketch): Observe that a user does not learn (g, b_i, e), which prevents him from finding the trapdoor value $x = g^{1/e}$. It is trivial to see that if the these values are known to two users, x can then be found: Assuming there are four forgers; each possesses a valid decryption key, (a_i, b_i), for $i = 1, \cdots, 4$. Given the first and second ones, we find they have the relation: $b_2/b_1 = (g^{a_1-a_2})^{1/e}$. Similarly, for the third and four ones, we have $b_4/b_3 = (g^{a_3-a_4})^{1/e}$. The forgers can then try to find two suitable integers α and β such that $\alpha(a_1 - a_2) + \beta(a_3 - a_4) = 1$. With the resulting α and β, they can compute $(b_2/b_1)^{\alpha}(b_4/b_3)^{\beta} = g^{1/e} = x$. Once x is found, they can compute other decryption keys.

Therefore, in our scheme (g, b_i, e) are not given to the users. With a similar attack to the above, at best the forgers can compute $(\hat{b}_2/\hat{b}_1)^C$ and $(\hat{b}_4/\hat{b}_3)^C$ which are equal to $g^{r(a_1-a_2)}$ and $g^{r(a_3-a_4)}$, respectively. Using the same approach, they obtain a pair of (α, β) and thus g^r. Since g^r is public, they gain nothing from it. □

4. The Scheme with IND-CCA2 Security

Soldera *et al.*[14] proposed an encryption scheme that is claimed having IND-CCA2 security under the DDH assumption. His scheme is a variant of Zheng-Seberry scheme [16], which is believed insecure under IND-CCA2 [14]. The reader is referred to [1] for more information. We have shown that our scheme is not based on the DDH but the RSA assumption. However, their algorithm can be converted into our scheme, although the security assumption is different.

The Setup and Subscribe of the scheme are the same. Here, we give the Encrypt and Decrypt phases:
Encrypt: Select random $r, \theta \in \mathbb{Z}_{\phi(N)}$, compute $u = y^r$, $v = H(M\|u)$, $w = M\|v$, $A = uw$, $B = g^r$, $C = r\theta$. The resulting ciphertext is (A, B, C).
Decrypt: Compute $A(B^a\hat{b}^C)^{-1} = M\|v' = w'$, and then verify the decryption by checking $H(M\|\frac{A}{w'}) = v'$

Theorem 4 *Our scheme is secure against the CCA2 attacks in the random oracle model assuming that the RSA assumption is intractable.*

Proof (Sketch): There exists a simulator that simulates the encryption oracle as follows. The simulator is different from the "standard" one which can randomly pick an encryption key. We assume that the encryption key is fixed to suit our scheme better. There exits a decryption oracle. The adversary can send any ciphertext, which might not be necessarily from the simulator, to the decryption oracle and obtain a pair (y_i, M_i).

On input two random messages $M_0, M_1 \in \mathbb{Z}_N^*$, the simulator outputs the encryption of these messages:

$$(u_0 y^{r_0}, g^{r_0}, r_0/\theta, u_0 = y^{r_0}, v_0 = H(M_0\|u_0), w_0 = M_0\|v_0) \leftarrow \mathsf{Encrypt}(m_i),$$
$$(u_1 y^{r_1}, g^{r_1}, r_1/\theta, u_1 = y^{r_1}, v_1 = H(M_1\|u_1), w_1 = M_1\|v_1) \leftarrow \mathsf{Encrypt}(M_{1-i}),$$
$$i \in_R \{0, 1\},$$

where $r_0, r_1 \in_R \mathbb{Z}_{\phi(N)}$. If the RSA assumption is intractable, then the adversary picks i randomly. The advantage of the adversary is $\Pr(\mathsf{win}) - \frac{1}{2}$.

The ciphertext strings are sent to the decryption oracle. If the RSA assumption is trackable to the adversary, the adversarial algorithm outputs a forgery: $(a', \hat{b}')$, $y^{r_0} = B^{a'}\hat{b'}^C$, and if the adversary chooses $i = 0$, then

$$u_i y^{r_0} = \begin{cases} y^{r_0} & (i = 0) \\ u_1 y^{r_0} & (i = 1) \end{cases}$$

The result is distinguishable to the adversary who is sure that the first ciphertext encrypts the first message with probability non-negligibly better than random guessing, because y_0^r is associated with the forged key $(a', \hat{b}')$ Of course, if the RSA problem is intractable, then the adversary cannot determine y^{r_0}. □

5. Conclusion

We propose an efficient broadcast encryption scheme based on the many trapdoor function by the assumption that the RSA problem is not solvable in polynomial time. Our scheme achieves the same goals given in the original Pay TV paper but is more efficient. We also described a variant of our scheme, which provides IND-CCA2 security.

References

[1] J. Baek and Y. Zheng. Zheng and Seberry's public key encryption scheme revisited. *International Journal of Information Security*, 2(1):37–44, 2003.

[2] M. Bellare, S. Halevi, A. Sahai, and S. Vadhan. Many-to-one trapdoor functions and their relation to public-key cryptosystems. In H. Krawczyk, editor, *Advances in Cryptology, Proc. CRYPTO 98,* LNCS 1462, pages 248–299. Springer Verlag, 1998.

[3] D. Boneh and X. Boyen. Efficient selective-id secure identity based encryption without random oracles. In *Advances in Cryptology, Proc. EUROCRYPT 2004,* LNCS 3027, pages 223–238. Berlin: Springer-Verlag, 2004.

[4] D. Boneh and M. Franklin. An efficient public key traitor tracing scheme. In *Adances in cryptology - CRYPTO '99, Lecture Notes in Computer Secience 1666*. Springer Verlag, 1999.

[5] R. Cramer and I. Damgard. New generation of secure and practical RSA-based signatures. In *Advances in Cryptology, Proc. CRYPTO 96,* LNCS 1109, pages 173–185. Springer-Verlag, Berlin, 1996.

[6] R. Cramer and V. Sharp. Signature schemes based on the strong RSA assumption. In *Proc. of 6th ACM conference on computer and communication security*, pages 173–185. Springer-Verlag, Berlin, 1999.

[7] A. Fiat and M. Naor. Broadcast encryption. In D. R. Stinson, editor, *Advances in Cryptology, Proc. CRYPTO 93,* LNCS 773, pages 480–491. Springer Verlag, 1994.

[8] E. Gafni, J. Staddon, and Y. Yin. Efficient methods for integrating traceability and broadcast encryption. In *Advances in Cryptology, Proc. CRYPTO 99,* LNCS 1666, pages 372–352. Springer Verlag, 1999.

[9] A. Garay, J. Staddon, and A. Wool. Long-lived broadcast encryption. In *Advances in Cryptology, Proc. CRYPTO 2000,* LNCS 1880, pages 333–352. Springer Verlag, 2000.

[10] R. Gennaro. Multi-trapdoor commitments and their applications to proofs of knowledge secure under concurrent man-in-the-middle attacks. In *Advances in Cryptology, Proc. CRYPTO 2004,* LNCS 3152, pages 220–236. Springer-Verlag, 2004.

[11] D. Halevy and A. Shamir. The LSD broadcast encryption scheme. In *Advances in Cryptology, Proc. CRYPTO 2002,* LNCS 2442, pages 47–60. Springer Verlag, 2002.

[12] M. Luby and J. Staddon. Combinatorial bounds for broadcast encryption. In *Advances in Cryptology, Proc. EUROCRYPT 98,* LNCS 1403, pages 512–526. Springer Verlag, 1998.

[13] A. Narayanan, C. P. Rangan, and K. Kim. Practical pay TV schemes. In *Information Security and Privacy–ACISP 2003,* LNCS 2727, pages 192–203. Springer Verlag, 2003.

[14] D. Soldera, J. Seberry, and C. Qu. Tha analysis of Zheng-Seberry scheme. pages 159–168. Springer Verlag, 2002.

[15] T. Tsiounis and M. Yung. On the security of ElGamal based encryption. pages 117–135. Springer Verlag, 1998.

[16] Y. Zheng and J. Seberry. Immunizing public key cryptosystems against chosen ciphertext attacks. *IEEE Journal on Selected Areas in Communications*, 11(5):715–724, 1993.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

A Generic Scheme for Zero-Knowledge Sets [1]

R. Xue [a,2], D. Song [b], Z. Zhang [a], and D. Feng [a]
[a] *State Key Lab. of Information Security, IOS, CAS*
[b] *ECE & CS Dept., Carnegie Mellon University*

Abstract. Zero-Knowledge sets, proposed by Micali *et al.* in FOCS'03, allow the owner of a set S to publish a very short commitment C_S to S, so that the owner can later prove or disprove, against C_S, the membership of any (potential infinity many) elements chosen by the verifier, without leaking more about S than the membership of the elements. This new secure primitive is proved to be useful in private data queries, and other similar scenarios where depends on the trust and privacy.

We investigate the theoretical primitives underline this new secure notion. The main contribution of this paper is to present a generic scheme for zero-knowledge sets which is as efficient as that in [1]. The new scheme is constructed by adopting the Merkle type of commitment under the assumption of existence of claw free pairs of trapdoor pseudo-permutations.

Keywords. zero-knowledge sets, pseudo permutation, Merkle tree, chameleon hash, trapdoor commitment

1. Introduction

The notion of Zero-Knowledge sets was recently proposed by Micali, Rabin, and Kilian in [1]. For any arbitrarily finite set S, the prover in a zero-knowledge scheme makes a commitment C_S to S. The challenger would ask (adaptively) a sequence of elements $x_1, x_2, \ldots$. The prover could give corresponding proof, against C_S, whether $x_i \in S$ or $x_i \notin S$ without revealing more knowledge about S than the membership of these elements. That is, the scheme should guarantee that S's cardinality remain hidden to the maximum extent possible.

Zero-knowledge proof of a membership is investigated in plenty of literatures (see, e.g. [2,3]). Zero-Knowledge sets is, however, not only membership problem, rather, a *decidable membership problem.* That means, it should not only give out the proof for any string in the set, but also for strings that are not in the set. The main difficulty here lies in proving of potentially infinity possible strings that do not lie in S.

[1]Supported respectively by NSFC under grant No.60373048, No.60273027 and No. 60373039.
[2]Correspondence to: Rui Xue, Institute of Software, CAS. Beijing, 100080. E-mail: xuerui@ios.cn.

Zero-Knowledge sets scheme should possess three secure requirements: completeness, soundness, and zero-knowledge. Given a secure parameter k, a complete secure zero-knowledge sets scheme should be valid for any finite set S. That is, to commit to S, and for any $x \in S$, correctly prove the membership, or not for $x \notin S$. The soundness guarantees that prover cannot lie about the membership of any element x. Zero-Knowledge guarantees that the knowledge obtainable by seeing a commitment to a set S and the sequence of proofs for the membership of $x, y, \ldots$, coincides with that obtainable without seeing anything about S at all, except for asking a trusted party about the membership of strings $x, y, \ldots$, and receiving in response his truthful but unproved assertions about the membership of them. For example $x \notin S, y \in S, \ldots$.

Micali *et al* in [1] proposed there , in fact, a more general notion: an efficient zero-knowledge Elementary Database (EDB for short) scheme based on the DDH assumption. Loosely speaking, an EDB is a partial function with a finite domain, which is easy to see to subsume zero-knowledge sets as a special case. Under the discrete logarithm assumption, by making use of Pedersen's commitment[4] as elementary commitment to the messages, they could construct an efficient zero-knowledge EDB scheme. We will present a general scheme for EDB in this paper, instead of only for zero-knowledge set.

The motivation here is to investigate the cryptographic primitive that is suffice for zero-knowledge EDB. The endeavor to find weakest primitive for a cryptographic object is popular and important. It is well known that one way function is necessary and sufficient for signature (e.g. see [5]), commitment, and the existence of pseudo-random generator [6]. The existence of trapdoor permutations implies secure public key encryption scheme.

Related Works Ostrovsky, Rackoff and Smith[7] investigated the consistent query protocol. The protocol there was modified to achieve the private property the same as the zero-knowledge set does. In order to reach this property, zero-knowledge proof was employed. It got a interactive proof instead of non-interactive like we do here.

Chase, Healy, Lysyanskaya, Malkin and Reyzin proposed a new kind of commitment so called *mercurial commitments* in [8], which is available for authors of this paper only after Eurocrypt'05, though it is announced on the web earlier. The new kind of commitment abstract the ideas from zero-knowledge , could be implemented based on various assumptions including claw-free functions, mainly from trapdoor functions. The paper uses also Merkle tree to construct zero-knowledge sets. Our result directly makes use of chameleon hash function and adopts the Merkle type of commitment to construct the scheme of zero-knowledge set.

Our Contributions In this work we present a scheme for zero-knowledge EDB under the existence pairs of claw-free trapdoor pseudo-permutations. The type of commitment to any given EDB D adopts Merkle type of commitment. First to commit to the value $D(x)$ of keys x in $[D]$ (the support set of function D) in Merkle tree, and then to put the value stored in the root as a commitment to D. Proof to the value of a key in $[D]$(the range of D) is an authentication procedure as usual in Merkle tree. The challenge is how to prove a key that is not in the support $[D]$ of D. A key point is to use at empty nodes in Merkle tree where any message can be stored. A trapdoor hash function is used there, instead of collision free hashing as for full nodes. To achieve the zero-knowledge property, we implement it with claw free trapdoor pseudo permutation pairs and a uniformity collision free hash function G. The commitment to the message stored at each node is a simple one: to commit to m, choose a uniformly random string r, compute $G(m, r)$ as the commitment to m. This way, we could get a perfect complete, sound and zero-knowledge scheme to EDB.

Organization of the paper: After an introduction to preliminary notions in section 2, we prove how to implement chameleon hash functions by clew free pairs of trapdoor pseudo permutation in section 3. Our generic scheme for zero-knowledge EDB and its proof of security are presented in section 4. Section 5 summarizes our main result. Due to the lack of space, some formal definition used in the paper are put as appendix A.

2. Primary Notions

Merkle Tree: When constructing a new digital signature scheme in [9], Merkle used a tree structure known as Merkle tree. Though Merkle tree could be implemented as arbitrary branch tree, it is usually with binary tree without losing efficiency. Let $\mathcal{B}_k$ be the complete binary tree with 2^k leafs, and each of its leaves is associated a name. The root has the name ε and its left son has the name 0, right son the name 1. Recursively, if a node has a name b, then its left son has name $b0$ and right son $b1$. A node is in the i'th *level* if and only if its name has i bits, or its distance from root is i. Let $S \subseteq \{0, 1\}^k$ be a subset of leafs in $\mathcal{B}_k$. Following the notions in [1], we define subtree $\texttt{Tree}(S)$ of $\mathcal{B}_k$ as the union of all paths induced by the leafs in S, that is, $\texttt{Tree}(S)$ is the subtree formed by all paths from root to nodes in S. Frontier of S is the set of all nodes in $\mathcal{B}_k$ that do not lie in $\texttt{Tree}(S)$ but being siblings of nodes in $\texttt{Tree}(S)$. Formally, $\texttt{Frontier}(S)$ is defined as

$$\{\, v \in \mathcal{B}_k \mid v \notin \texttt{Tree}(S),\ v\text{'s sibling is in } \texttt{Tree}(S) \,\}$$

In order to make a subtree $\texttt{Tree}(S) \cup \texttt{Frontier}(S)$ into a **Merkle tree** M, we store values in each of nodes in the subtree as following: firstly, to store any binary string in any childless nodes. Then, assuming H a collision-free hash function, for any node v in $\texttt{Tree}(S) \cup \texttt{Frontier}(S)$ with left son $v0$ and right son $v1$, stored V_{v0} and V_{v1} respectively, to store $H(V_{v0}V_{v1})$ in v. The string stored in the root ε is the commitment of all values stored in M's nodes. To prove that a node v stored a value d, one presents a proof sequence consisting of all the values stored in the path from root to node v and those of their siblings. A fake proof would result in a collision of H.

Zero-Knowledge Sets or EDB: The notion of zero-knowledge sets as a new primitive is proposed by Micali, Rabin and Kilian in [1]. It is about a commitment to a finite set S that allows the polynomial time prover, later on, could reveal a proof for any string x, whether $x \in S$ or $x \notin S$ without revealing any knowledge beyond the validity of these membership assertions.

Zero-knowledge set is a special case of Zero-Knowledge Elementary Database (EDB for short). EDB is essentially a finite function D that maps finite *keys* $[D]$ into *values*, where $[D]$ is the range of D. For each key x, we obtain $\perp$ or a value $D(x)$. Zero-Knowledge EDB consists of a commitment to a EDB D and for any $x \in \{0,1\}^*$, a proof that x is not a key in D, or x's value is $D(x)$ without revealing any undue knowledge. The formal definition from [1] is omitted here (A formal definition is appeared in appendix for convenience).

As was pointed out by Micali *et al* [1] that difficulty for a scheme of zero-knowledge sets lies in the proving potentially infinite possible the values for $x \notin [D]$. They proposed an efficient scheme by presenting a commitment dependant on Pederson commitment [4]. We could here construct a scheme with generic cryptographic primitive, which as efficient as the one in [1].

Secure Commitment Scheme The noninteractive commitment constitutes of two phases: commit and verification phase. In the commit phase, committer computes strings (c, d) against the given message m. Where c is the commitment to m to be sent to verifier, and d the proof that will be kept secret by committer himself. In the verifying phase, committer sends d to verifier, the latter uses d to de-commit the c to check its validity against m.

For any collision free hash function $h\colon \{0,1\}^* \to \{0,1\}^k$, any given message $m \in \{0,1\}^*$, uniformly random chosen $r \in \{0,1\}^*$, let $c = H(m, r)$, $d = r$. Then we call (c, d) is a **simple commitment** to m.

3. Chameleon Hash Functions Ensemble

Chameleon hash function, introduced by Krawczyk and Rabin in [10], is a type of trapdoor and collision resistant hash function. Given any input, one can compute the output of the hash function effectively, and it is infeasible (without information of trapdoor) to find its collision, namely to find out two input which maps to the same output. However, with the knowledge of the trapdoor, one could easily find collisions for every given input. The formal definitions are given in the appendix.

For our purpose in the construction of the new scheme for zero-knowledge sets, it is enough to use chameleon hash family S_n^m in which each function maps n-bit strings to m-bit strings. Reader may consult, for example, [2] section 3.5.1.1 for the definition of hashing family S_n^m. Where strings are also used for the descriptions of functions as we do here.

In the following, we assume that for any different messages m_1, m_2, they are non-prefix. That is none of them is a prefix of the other. This could be obtained by transformation of any massages $m = m_1 m_2 \ldots m_t$ into $m_1 0 m_2 0 \ldots m_t 1$. Then any different messages have non-prefix property after transformation. We will assume this non-prefix property without explicitly stating hereafter in the paper.

Lemma 3.1 *The existence of claw-free pairs of trapdoor pseudo-permutations implies the existence of chameleon hash functions.*

Proof of Lemma 3.1: Suppose $\{ (f_i^0, f_i^1) \mid i \in I \}$ is a collection of claw-free pairs of trapdoor pseudo-permutations. Use the same construction as in [10], following [11]: for any $i \in I$

```
Construction of $CH_i$:
Input $m, r \in \{0,1\}^*$, where $m = m_1 m_2 \cdots m_k$.
      $m_j \in \{0,1\}$, $1 \le j \le k$.
Output
$CH_i(m,r) = f_i^{m_k}(f_i^{m_{k-1}}(\cdots(f_i^{m_2}(f_i^{m_1}(r)))\cdots))$.
```

We show that $\{ CH_i \mid i \in I \}$ is a collection of chameleon functions. Conditions 1 and 4 in definition A.5 are obviously hold from that $\{ (f_i^0, f_i^1) \mid i \in I \}$ is a collection of claw-free pairs of trapdoor pseudo-permutations. The proof for condition 2 is similar to that for theorem 1 in [12], which we omit here.

To show the third condition, we denote g_i^l as the pre-image finder for f_i^l where $(l \in \{0,1\})$, described in 3, Definition A.4. Thus with trapdoor (i, t^l) for f_i^l, we have $\forall x \in \mathcal{D}_i$, $f_i^l(g_i^l(t^l, f_i^l(x))) = f_i^l(x)$. Let $t = (t_i^0, t_i^1)$ as the trapdoor, define $TF_i(t, m, r, m')$ as

$$g_i^{m'_1}(t^{m'_1}, \cdots (t^{m'_{k'-1}}, g_i^{m'_{k'}}(t^{m'_{k'}}, CH_i(m,r))) \cdots)$$

where $m' = m'_1 m'_2 \cdots m'_{k'}, m'_j \in \{0,1\}, 1 \le j \le k'$. It is easy to verify that

$$CH_i(m,r) = CH_i(m', TF_i(t, m, r, m'))$$

and TF_i is polynomial time because g_i^0, g_i^1 are. This complete the proof of the lemma. □

Lemma 3.2 *Suppose f is a chameleon function, g a collision-free function, and h any permutation, then $h \circ f$ is a chameleon function and $h \circ g$ is collision-free.*

The following result will be used in next section:

Lemma 3.3 *Let $\{ F_i \mid i \in I \}$ be a pseudo-random permutation ensemble, $G \colon \{0,1\}^* \to \{0,1\}^k$ is a collision-free hash function so that, for any $m \in \{0,1\}^*$, $r \overset{R}{\leftarrow} \{0,1\}^*$, distribution $G(m,r)$ induced by the uniformly random string r is uniform on $\{0,1\}^k$ if r is uniform chosen in $\{0,1\}^*$. Then $F = \{ F_i \circ G \mid i \in I \}$ is a pseudo-random collision-free ensemble. In the following of this paper we call each function in F* **a variant of** *G.*

Proof is, making use the result in lemma, by hybrid arguments and omitted.

4. A Generic Scheme for ZKEDB

4.1. Basic Preparation

In the scheme of zero-knowledge EDB, the committer and the verifiers share a public and random reference string σ of length polynomial in k, where k is secure parameter. The committer could create a collision free hash function G from σ so that $G\colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^k$, and for any $m \in \{0,1\}^*$, $G(m,r)$ is uniformly distributed on $\{0,1\}^k$ if r is uniformly chosen from $\{0,1\}^*$. This hash function can be uniquely extracted by any deterministic ptime (polynomial time) machine. We will use another hash function H to form Merkle tree, which is uniquely extracted from σ.

In a commitment, committer chooses a chameleon function ensemble as in definition A.5

$$CH = \{ f_i \mid f_i\colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^k, i \in I \}$$

and an ensemble F of permutation variant of G as stated in 3.3.

Given an EDB $D = \{ (x_i, y_i) \mid i \in I \}$, the committer uses H to hash x_i into k bit string, so that $H(x_i)$ is a leaf of $\mathcal{B}(k)$. We will assume in the following that for any given EDB D, each string in the support $[D]$ of D is of length k, formally, $[D] \subseteq \{0,1\}^k$. For any string x inquired by verifier during the proving phase in the following, prover will first hash it with G into a k bit string, and then give a proof of its value. This procedure will not be explicitly stated in next sections.

4.2. Commitment of an **EDB**

With the Merkle tree $\mathtt{T}(D)$ constructed for any given **EDB** D as stated in section 2, each of its nodes will be associated or assigned (stored) some values and, then recursively to form a Merkle type of commitment for **EDB** D. The collision free hash function $H\colon \{0,1\}^* \to \{0,1\}^k$ generated during the preparation phase is used to form the desired Merkle commitment, and the fixed one-way collision free function $G\colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^k$ and its variants associated to the nodes of the tree. The variants of G are results of composition from G with pseudo-random permutations g as stated in lemma 3.3.

Note that tree $\mathtt{T}(D) = \mathtt{Tree}(D) \cup \mathtt{Frontier}(D)$, the nodes in $\mathtt{Tree}(D)$ are called *full* nodes, while those in $\mathtt{Frontier}(D)$ *empty* nodes. Each full node α is associated with a hash function f_α that is a variant of G and an uniformly random string in $r_\alpha \in \{0,1\}^*$, and is stored two values: one is formed according to Merkle commitment, the other is evaluated with the first value and its associations. Each empty node β in $\mathtt{T}(D)$ is associated with a chameleon hash function g_β that is indistinguishable with G and an uniformly random string in $r_\beta \in \{0,1\}^*$. A three phases commitment process is as follows:

To assign values for leaf: There are two kinds of leafs in $\mathtt{T}(D)$: the full ones and empty ones.
Each full leaf α corresponding to a point x in the support $[D]$ is assigned $m_\alpha = G(y)$, if $D(x) = y$. All empty leaf β in $\mathtt{T}(D)$ is assigned $m_\beta = 0$.

Association to nodes: An uniformly random string $r_\alpha \overset{R}{\leftarrow} \{0,1\}^*$ is associated to node α. The committer associates each full node α a variant f_α of G, where $f_\alpha = g_\alpha \circ G$ and g_α is a random permutation. Each empty node β associates a chameleon hash function f_β.

Merkle-Commitment: After the assignments and associations as above, the Merkle type of commitment for $\mathtt{T}(D)$ is computed recursively as follows:

- If node α is a leaf in $\mathtt{T}(D)$, to compute $c_\alpha = f_\alpha(m_\alpha, r_\alpha)$ and store (c_α, f_α).
- If the internal node α associates to f_α, with left son $\alpha 0$ and right son $\alpha 1$ which have already assigned with values $m_{\alpha 0}$ and $m_{\alpha 1}$ respectively, and stored $c_{\alpha 0}, f_{\alpha 0}$ and $c_{\alpha 1}, f_{\alpha 0}$ respectively. α is assigned the message

$$m_\alpha = H(c_{\alpha 0}, c_{\alpha 1}, f_{\alpha 0}, f_{\alpha 1}) \tag{1}$$

And is stored with value:

$$c_\alpha = f_\alpha(m_\alpha, r_\alpha) \tag{2}$$

(c_α, f_α) is called α's *identifier*, denoted as $\mathtt{ID}_\alpha$.

The **commitment to an EDB** D is the root's identifier $\mathtt{ID}_\varepsilon$. The result tree is called an assigned tree of D, denoted as $\mathtt{AT}(D)$. Please note that an assigned tree $\mathtt{AT}(D)$ consists of the tree $\mathtt{T}(D)$, together with the identifiers for all nodes, and the assigned values for leafs.

Definition 4.1 Suppose $\alpha_1, \ldots, \alpha_s, \alpha_{s+1}$ is a path in assigned tree $\mathtt{AT}(D)$ from leaf α_1 to root $\alpha_{s+1} = \varepsilon$, and $\beta_1, \ldots, \beta_s$ are corresponding sibling sequence, that is β_i is the sibling of α_i.

1. The sequence $\alpha_1, \beta_1, \ldots, \alpha_s, \beta_s, \alpha_{s+1}$ is called the **authentication sequence** for α_1 (and for β_1), and a tuple sequence $\mathtt{ID}_{\alpha_1}, \mathtt{ID}_{\beta_1}, \ldots, \mathtt{ID}_{\alpha_s}, \mathtt{ID}_{\beta_s}, \mathtt{ID}_{\alpha_{s+1}}$ is called **a proof sequence** for α_1 (and for β_1). Where each $\mathtt{ID}_\gamma$ is an ordered pair (c, f) with c being strings and f a function (also a binary string).
2. A proof sequence $\mathtt{ID}_{\alpha_1}, \mathtt{ID}_{\beta_1}, \ldots, \mathtt{ID}_{\alpha_s}, \mathtt{ID}_{\beta_s}, \mathtt{ID}_{\alpha_{s+1}}$ is called $H-$**consistent** (or shortly, **consistent**) if the corresponding equations (1) and (2) hold against an aiding sequence $r_{\alpha_1}, r_{\beta_1}, \ldots, r_{\alpha_s}, r_{\beta_s}, r_{\alpha_{s+1}}$.

4.3. Assigned Merkle tree for an EDB

In this section we expand the assigned tree $\mathtt{AT}(D)$ for an **EDB** D into a complete binary tree, and to make it a complete assigned Merkle tree. Please note that we introduce here the way how to weld them together, but it will practically be computed on-the-fly during the enquiry as remarked at the end of this section.

Let k be the secure parameter, then tree $\mathtt{T}(D)$ is a subtree of the complete binary tree $\mathcal{B}_k$ of depth k. Notice that to make a complete assignment tree, it only need to assign all the subtrees that is rooted by the empty nodes of $\mathtt{T}(D)$, and then to weld them into $\mathtt{AT}(D)$.

Suppose β is an empty node in assigned tree $\mathtt{AT}(D)$, it has been already assigned with $m_\beta = 0$, with identifier $\mathtt{ID}_\beta = (c_\beta, f_\beta)$ and associated random r_β. If β is a leaf in $\mathcal{B}_k$, then nothing to do. Else, let $\mathtt{T}_\beta$ is the subtree in $\mathcal{B}_k$ rooted by β. For any leaf γ in $\mathtt{T}_\beta$, let $m_\gamma = 0$, f_γ be a chameleon hash function uniformly random picked in CH, r_γ be an uniformly random string in $\{0,1\}^*$, let $c_\gamma = f_\gamma(m_\gamma, r_\gamma)$. Then its identifier is (c_γ, f_γ).

After all the leafs of T_β have been assigned, internal nodes could be processed in a recursive bottom-up fashion: first to assign a random (chameleon) hash function and a random string for each internal node in $\mathtt{T}_\beta$, then to evaluate the messages m and commitments c following (1) and (2).

The recursive procedure finally comes to the root β. The originate identifier $\mathtt{ID}_\beta = (c_\beta, f_\beta)$ should be kept in order to make the welding point smooth. Now, let $m'_\beta = H(c_{\beta 0}, c_{\beta 1}, f_{\beta 0}, f_{\beta 1})$. To keep the assignment consistent, recalling that f_β is a chameleon hash function and by using the trapdoor information, one could find an $r' \in \{0,1\}^*$ such that $f_\beta(m_\beta, r_\beta) = c_\beta = f_\beta(m'_\beta, r'_\beta)$. This way, with the aid of the new random string r', node β's identifier is kept to be $\mathtt{ID}_\beta = (c_\beta, f_\beta)$. From (2) and the way of of our assignment we know that this new ID will not affect the ID of β's father node. Thus the subtree $\mathtt{T}_\beta$ is weld with $\mathtt{AT}(D)$. After welding of all empty nodes rooted subtrees we get a completely assigned tree $\mathcal{B}_k(D)$. The following result is easy to see

Lemma 4.2 *For all $x \in \{0,1\}^*$, there exists a consistent proof sequence for node $H(x)$ by $\mathcal{B}_k(D)$ with $m_{H(x)} = G(y)$ if $D(x) = y$, or $m_{H(x)} = 0$ if $D(x) = \perp$.*

Remark Though we have introduced how to weld an assigned tree $\mathtt{AT}(D)$ into a binary tree $\mathcal{B}_k(D)$. It is, due to the efficiency consideration, not necessary to weld an assigned tree into a complete assigned Merkle tree during the commitment phase. That is, committer only keep relative values in $\mathtt{AT}(D)$. In proof phase, prover (committer) will compute the corresponding values along the path from the inquired key (that not in $\mathtt{AT}(\mathrm{D})$) to the root and the value of their siblings when necessary (see next section for details).

4.4. Proving the values of an EDB

The prover and verifier will share an uniformly random string $\sigma \in \{0,1\}^{k^c}$, where c is some positive real constant.

Given any **EDB** D, the prover commits the **EDB** D as previous sections and publishes the final commitment $\mathtt{ID}_\varepsilon$ for D. He keeps privately in memory those associated (or assigned) to and stored in the nodes of $\mathcal{B}_k(D)$.

For any given value x, prover has to give out a proof of $D(x) = y$ if $x \in [D]$, or a proof of $D(x) = \perp$ if not. With the completely assigned tree $\mathcal{B}_k(D)$ in mind, prover could process it as follows.

Proving $D(x) = y$. Prover finds the path $\alpha_1, \alpha_2, \ldots, \alpha_k, \varepsilon$ from leaf $H(x)$ to root ε in $\mathcal{B}_k(D)$, and the corresponding sibling sequence $\beta_1, \beta_2, \ldots, \beta_k$. Where $\alpha_1 = H(x)$.

1. Prover reveals to verifier y, and a proof sequence

$$\mathtt{ID}_{\alpha_1}, \mathtt{ID}_{\beta_1}, \mathtt{ID}_{\alpha_2}, \mathtt{ID}_{\beta_2}, \ldots, \mathtt{ID}_{\alpha_k}, \mathtt{ID}_{\beta_k}, \mathtt{ID}_\varepsilon$$

and together with an aiding sequence and a permutation function sequence as witness

$$r_{\alpha_1}, r_{\beta_1}, \ldots, r_{\alpha_k}, r_{\beta_k}, r_{\alpha_\varepsilon} \qquad g_{\alpha_1}, g_{\alpha_2}, \ldots, g_{\alpha_k}, g_\varepsilon$$

2. Verifier checks that the validity of $f_{\alpha_i} = g_{\alpha_i} \circ G$ for all $1 \leq i \leq k$, and $f_\varepsilon = g_\varepsilon \circ G$. If any of them fails, then **reject**. Else, verifier checks if $c_{\alpha_1} = f_{\alpha_1}(G(y), r_{\alpha_1})$ and whether proposed proof sequence is consistent against the aiding sequence or not. He **accepts** when both are true, else **rejects**.

Verifier will be convinced that the validity of value of x is $D(x)$, for G is a collision free hash, and the way of Merkle authentication is convincing.

Proving $D(x) = \perp$. This case is similar to the last one and even simpler. Prover finds the path from leaf $H(x)$ to root ε in $\mathcal{B}_k(D)$, denoted as $\alpha_1, \alpha_2, \ldots, \alpha_k, \varepsilon$, and finds the corresponding sibling sequence $\beta_1, \beta_2, \ldots, \beta_k$.

1. Prover reveals to verifier $\perp$, and a proof sequence

$$\mathtt{ID}_{\alpha_1}, \mathtt{ID}_{\beta_1}, \mathtt{ID}_{\alpha_2}, \mathtt{ID}_{\beta_2}, \ldots, \mathtt{ID}_{\alpha_k}, \mathtt{ID}_{\beta_k}, \mathtt{ID}_\varepsilon$$

2. Verifier checks whether $c_{\alpha_1} = f_{\alpha_1}(0, r_{\alpha_1})$ and whether proposed proof sequence is consistent or not. He **accepts** when both are true, else **rejects**.

Unlike last case for $x \in [D]$, prover could now give infinite possible potential proof sequences for any $x \notin [D]$, that is $D(x) = \perp$. The reason is that prover assigns chameleon hash functions for some of the nodes.

4.5. Secure Properties of the Scheme

Any secure scheme for **EDB** should enjoy three properties: perfect completeness, soundness and zero-knowledge.

Completeness is obvious from our construction.

Soundness is guaranteed by showing that prover cannot cheat. That is, it is hard for prover to give out different values of proof sequences for any giving x. We have the following result

Lemma 4.3 *It is hard for prover to present two valid proof sequences simultaneously for proving*

1. $D(x) = y$ *and* $D(x) = y'$ *with* $y \neq y'$, *or*
2. $D(x) = y$ *and* $D(x) = \perp$.

that both are accepted by verifier.

Proof of Lemma 4.3: We only show Lemma for the first case, that for the second case is similar and is thus omitted.

Let $\alpha_1, \beta_1, \ldots, \alpha_k, \beta_k, \alpha_\varepsilon$ be $H(x)$'s authentication sequence in $\mathcal{B}_k(D)$. Suppose, to the contradictory, prover presents two acceptable proof sequences for $D(x) = y$ and $D(x) = y'$ as follows:

1. Prover presents a proof for $D(x) = y$ by giving the following:

$$\mathtt{ID}_{\alpha_1}, \mathtt{ID}_{\beta_1}, \mathtt{ID}_{\alpha_2}, \mathtt{ID}_{\beta_2}, \ldots, \mathtt{ID}_{\alpha_k}, \mathtt{ID}_{\beta_k}, \mathtt{ID}_\varepsilon$$

and together with an aiding sequence and a permutation function sequence as witness

$$r_{\alpha_1}, r_{\beta_1}, \ldots, r_{\alpha_k}, r_{\beta_k}, r_{\alpha_\varepsilon} \qquad g_{\alpha_1}, g_{\alpha_2}, \ldots, g_{\alpha_k}, g_\varepsilon$$

And simultaneously,

2. Giving a proof for $D(x) = y'$ by giving:

$$\mathtt{ID}'_{\alpha_1}, \mathtt{ID}'_{\beta_1}, \mathtt{ID}'_{\alpha_2}, \mathtt{ID}'_{\beta_2}, \ldots, \mathtt{ID}'_{\alpha_k}, \mathtt{ID}'_{\beta_k}, \mathtt{ID}'_\varepsilon$$

and together with an aiding sequence and a permutation function sequence as witness

$$r'_{\alpha_1}, r'_{\beta_1}, \ldots, r'_{\alpha_s}, r'_{\beta_s}, r'_{\alpha_{s+1}} \qquad g'_{\alpha_1}, g'_{\alpha_2}, \ldots, g'_{\alpha_k}, g'_\varepsilon$$

That both are accepted implies that they are consistent proof sequences and $\mathtt{ID}_\varepsilon = \mathtt{ID}'_\varepsilon$. Let s be the minimum i such that $\mathtt{ID}_{\alpha_i} = \mathtt{ID}'_{\alpha_i}$, where $1 \le i \le k+1$, where $\alpha_{k+1} = \varepsilon$.

- Case $s > 1$. Assume w.l.o.g. that $\alpha_{s-1} = \alpha_s 0, \beta_{s-1} = \alpha_s 1$. By assignment, we have

$$m_{\alpha_s} = H(c_{\alpha_s 0}, c_{\alpha_s 1}, f_{\alpha_s 0}, f_{\alpha_s 1}), \quad c_{\alpha_s} = f_{\alpha_s}(m_{\alpha_s}, r_{\alpha_s}) \tag{3}$$

and

$$m'_{\alpha_s} = H(c'_{\alpha_s 0}, c'_{\alpha_s 1}, f'_{\alpha_s 0}, f'_{\alpha_s 1}), \quad c'_{\alpha_s} = f'_{\alpha_s}(m'_{\alpha_s}, r'_{\alpha_s}) \tag{4}$$

  - $*$ If $m_{\alpha_s} \neq m'_{\alpha_s}$, then by $(c_{\alpha_s}, f_{\alpha_s}) = \mathtt{ID}_{\alpha_s} = \mathtt{ID}'_{\alpha_s} = (c'_{\alpha_s}, f'_{\alpha_s})$, which means

$$c_{\alpha_s} = f_{\alpha_s}(m_{\alpha_s}, r_{\alpha_s}) = f'_{\alpha_s}(m'_{\alpha_s}, r'_{\alpha_s}) = c'_{\alpha_s}$$

    Prover finds a collision of f_{α_s} which is a variant of collision resistant hash function G, that contradicts assumption.
  - $*$ If $m_{\alpha_s} = m'_{\alpha_s}$, from the minimum of s, we know that $\mathtt{ID}_{\alpha_{s-1}} \neq \mathtt{ID}'_{\alpha_{s-1}}$, that is $c_{\alpha_s 0} \neq c'_{\alpha_s 0}$ or $f_{\alpha_s 0} \neq f'_{\alpha_s 0}$. Either case, from (3) and (4), would mean prover have found a collision of H.

- Case $s = 1$. We have now that $c_{H(x)} = c'_{H(x)}, f_{H(x)} = f'_{H(x)}$. Those imply that

$$\begin{aligned} f_{H(x)}(m_{H(x)}, r_{H(x)}) &= f'_{H(x)}(m'_{H(x)}, r'_{H(x)}) \\ &= f_{H(x)}(m'_{H(x)}, r'_{H(x)}) \end{aligned} \tag{5}$$

Because $f_{H(x)}$ is proven to be an variant of G by verifier, it is hard for prover to find $m_{H(x)} \neq m'_{H(x)}$ to satisfy equation (5). While $m_{H(x)} = m'_{H(x)}$ implies $G(y) = G(y')$ from the assignment, that contradicts to the assumption that G is collision resistant.

This ends the proof of lemma. □

Zero-Knowledge: Recall that the zero-knowledge EDB is to keep privacy of the membership of a new element in secret, it is easy to see from the variation of G that our scheme is zero-knowledge. Please note that it is unfeasible for a verifier in the scheme to determine that an element f_{α_i} taken from a membership proof is a chameleon hash function or not, this is due to the uniformness of chameleon function (see definition).

A note deserve to mention that it is usually not zero-knowledge for any hash functions with this scheme as pointed out in, e.g, [7], since the hash function would leak some information of the original messages. That is the reason why we use variation ensemble of G here instead of a single hash function. The zero-knowledge property guaranteed by the pseudo-randomness and the security of chameleon hash functions.

To summarize, we have given an efficient scheme of zero-knowledge EDB based on the existence of claw free pair of trapdoor pseudo permutations.That is,

Theorem 4.4 *Existence of claw free pairs of trapdoor pseudo permutations implies existence of zero-knowledge EDB. In this case we have an efficient construction of zero-knowledge EDB that as efficient as those in [1].*

5. Conclusion

In this paper we investigate the theoretical cryptography primitive underline zero-knowledge set, a novel cryptographic primitive proposed by Micali *et al* in [1]. Employing directly of chameleon functions which in turn is implemented by claw-free pair of trapdoor pseudo permutations, we are able to give a efficient construction of zero-knowledge EBD scheme. The more interesting work is to construct a scheme of zero-knowledge set under more weaker assumption.

Acknowledgements

We would like to thank anonymous reviewers of IWAP05 for comprehensive comments on this paper.

References

[1] Silvio Micali, Michael Rabin, and Joe Kilian. Zero-knowledge sets. In *Proc. 44th IEEE Symp. on Foundations of Comp. Science*, page 80, 2003.

[2] Oded Goldreich. *Foundations of Cryptography—Basic Tool*, volume I. Cambridge University Press, 2001.

[3] Oded Goldreich, Silvio Micali, and Avi Wigderson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*, 38(1):691-729, 1991.

[4] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In J. Feigenbaum, editor, *Advances in Cryptology, Proc. CRYPTO 91,* LNCS 576, pages 129–140, 1992

[5] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. 22nd ACM Symp. on Theory of Computing*, pages 387–394, Baltimore, Maryland, 1990. ACM.

[6] Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions. In *Proc. 21st ACM Symp. on Theory of Computing*, pages 12–24, Seattle, 1989. ACM.

[7] Rafail Ostrovsky, Charles Rackoff, and Adam Smith. Efficient consistency proofs for generalized queries on a committed database. In *ICALP*, pages 1041–1053, 2004.

[8] Melissa Chase, Alexander Healy, Anna Lysyanskaya, Tal Malkin, and Leonid Reyzin. Mercurial commitments with applications to zero-knowledge sets. In Ronald Cramer, editor, *EUROCRYPT'05*, volume 3494 of *Lecture Notes in Computer Science*, pages 422–439. Springer, 2005.

[9] R. C. Merkle. A digital signature based on a conventional encryption function. In *CRYPTO87*, volume 293 of *Lecture Notes in Computer Science*, pages 369–378, 1987.

[10] Hugo Krawczyk and Tal Rabin. Chameleon hashing and signatures. In *NDSS'2000*. Internet Society, 2000.

[11] I. B. Damgård. On the randomness of Legendre and Jacobi sequences. In S. Goldwasser, editor, *Advances in Cryptology, Proc. CRYPTO 88,* LNCS 403, pages 163–172, 1988.

[12] Alexander C. Russell. Necessary and sufficient conditions for collision-free hashing. In Ernest F. Brickell, editor, *Advances in Cryptology, Proc. CRYPTO 92,* LNCS 740, pages 433–441, 1992

A. Some Formal Definitions

Definition A.1 Let P_1, P_2, V be probability polynomial time (PPT) Turing machines. We denote PPT as the class of all PPT machines. k is security parameter, σ a binary string called the (random) reference string. The triple (P_1, P_2, V) constitute **a EDB system** if the machines do not retain state information after an execution and it processes as follows:

- *The committer* P_1 only produces public key PK_D and secret key SK_D for any D. That is $(PK_D, SK_D) \overset{R}{\leftarrow} P_1(D, 1^k, \sigma)$.
- *The prover* P_2 on input $(D, 1^k, \sigma, PK_D, SK_D)$ and additional input $x \in \{0,1\}^*$, computes a proof π_x. That is $\pi_x \overset{R}{\leftarrow} P_2(D, 1^k, \sigma, PK_D, SK_D, x)$.
- *The verifier* V to check the proof π_x for x. That is $r \overset{R}{\leftarrow} V(1^k, \sigma, PK_D, x, \pi_x)$, where $r \in \{D(x), out, \perp\}$. If $r = \perp$ then verifier believes the proof is false. $r = out$ means x is not in D's key.

The secure demanding here for an EDB is completeness, soundness and zero-knowledge. Following [1], we have:

Definition A.2 Let (P_1, P_2, V) be an EDB system. To say (P_1, P_2, V) is a zero-knowledge EDB if there exists a positive c such that

- *Perfect Completeness.* For all database D and $\forall x \in [D]$, the following probability is 1:

$$\Pr\left[\begin{array}{l}\sigma \overset{R}{\leftarrow} \{0,1\}^{k^c}; (PK_D, SK_D) \overset{R}{\leftarrow} P_1(1^k, \sigma, D);\\ \pi_x \overset{R}{\leftarrow} P_2(D, 1^k, \sigma, PK_D, SK_D, x):\\ V(1^k, \sigma, PK_D, x, \pi_x) = D(x)\end{array}\right]$$

- *Soundness.* $\forall x \in \{0,1\}^*$ and $\forall P' \in PPT$, the following is negligible

$$\Pr\left[\begin{array}{l}\sigma \overset{R}{\leftarrow} \{0,1\}^{k^c}; (PK', \pi_1', \pi_2') \overset{R}{\leftarrow} P'(1^k, \sigma):\\ V(1^k, \sigma, PK', x, \pi_1'), V(1^k, \sigma, PK', x, \pi_2') \neq \perp\\ \wedge V(1^k, \sigma, Pk', x, \pi_1') \neq V(1^k, \sigma, PK', x, \pi_2')\end{array}\right]$$

- *Zero-knowledge.* There exists a database simulator SIM such that for all Turing machine Adv, all integer $k > 0$, database D: $View(k) \approx View(k')$. Where

$$\begin{aligned} View(k) = \{\sigma \overset{R}{\leftarrow} \{0,1\}^{k^c};\\ &(PK_D, SK_D) \overset{R}{\leftarrow} P_1(1^k, \sigma, D);\\ &(x_1, s_1) \overset{R}{\leftarrow} Adv(1^k, \sigma, PK_D);\\ &\pi_{x_1} \overset{R}{\leftarrow} P_2(x_1, SK_D);\\ &(x_2, s_2) \overset{R}{\leftarrow} Adv(1^k, \sigma, PK_D, s_1, \pi_{x_1});\\ &\pi_{x_2} \overset{R}{\leftarrow} P_2(x_2, SK_D);\\ &\cdots : PK_D, x_1, \pi_{x_1}, x_2, \pi_{x_2}, \ldots\}\end{aligned}$$

$$\begin{aligned} View'(k) = \{(\sigma', PK', SK') \overset{R}{\leftarrow} SIM(1^k);\\ &(x_1, s_1) \overset{R}{\leftarrow} Adv(1^k, \sigma, PK');\\ &\pi_{x_1}' \overset{R}{\leftarrow} SIM^D(SK', x_1);\\ &(x_2, s_2) \overset{R}{\leftarrow} Adv(1^k, \sigma, PK', s_1, \pi_{x_1});\\ &\pi_{x_2}' \overset{R}{\leftarrow} SIM^D(SK', x_2);\\ &\cdots : PK', x_1, \pi_{x_1}', x_2, \pi_{x_2}', \cdots\}\end{aligned}$$

Here the $\approx$ is computational indistinguishability which denote computational zero-knowledge.

Definition A.3 [Collection of Claw-Free Pairs of Functions] A collection of claw-free pairs of functions is a collection of function tuples $\{ (f_i^0, f_i^1) \mid i \in I \}$ for some index set $I \subseteq \{0,1\}^*$ where $f_i^j : \mathcal{D}_i \to \mathcal{D}_i$ for some $\mathcal{D}_i \subseteq \{0,1\}^*$ such that

1. *Easy to sample and compute:* There exists an index generating algorithm $G \in PPT$ such that $G(1^n) \in \{0,1\}^n \cap I$. There exists a sampling algorithm S so that $S[i]$ is the uniform distribution on $\mathcal{D}_i$. There exists an evaluating algorithm $E \in PPT$ so that for all $i \in I, j \in \{0,1\}$, and $x \in \mathcal{D}_i, E[i,j,x] = f_i^j(x)$.
2. *claw-free:* For all claw finding algorithms $A \in PPT$, for all polynomial $p(\cdot)$, $\exists n_0, \forall n > n_0$,

$$\Pr\left[\begin{array}{r} i \leftarrow G[1^n], (x,y) \leftarrow A[i] : \\ f_i^0(x) = f_i^1(y) \end{array}\right] < \frac{1}{p(n)}$$

A collection of such functions is called **simple** if $\forall i \in I, \mathcal{D}_i = \{0,1\}^{|i|}$.

If, in addition, $f_i^j, j \in \{0,1\}$ in last definitions are all permutations on $\mathcal{D}_i$, then the result collection is a claw-free pairs of permutations. We, following Russell [12], extend it to be a collection of pseudo permutations. Loosely speaking, a pseudo-permutation is a function that is perhaps not a permutation but hard to find its collapses. The trapdoor pseudo-permutation f_i is a pseudo-permutation with extra (trapdoor) information $t(i)$ such that given any image $f_i(x)$ and trapdoor $t(i)$, it is easy to find a pre-image y such that $f_i(y) = f_i(x)$.

Definition A.4 [Collection of trapdoor Pseudo-Permutations] A collection of pseudo permutations is a collection of functions $\{ f_i \mid i \in I \}$ for some index $I \subseteq \{0,1\}^*$ where $f_i : \mathcal{D}_i \to \mathcal{D}_i$ for some $\mathcal{D}_i \subseteq \{0,1\}^*$ such that the following two conditions hold:

1. *Easy to sample and compute:* There exists a generating algorithm $G \in PPT$, a sampling algorithm S, and a evaluation algorithm $E \in PPT$, so that $G[1^n] \in \{0,1\}^n \cap I$. $\forall i, S[i]$ is the uniform distribution on $\mathcal{D}_i$. For all $x \in \mathcal{D}_i, E[i,x] = f_i(x)$.
2. *Collapse-free:* For all PPT collapse finding algorithms A, for all polynomial $p(\cdot)$, $\exists n_0, \forall n > n_0$

$$\Pr\left[\begin{array}{r} i \leftarrow G[1^n], (x,y) \leftarrow A[i] : \\ f_i(x) = f_i(y) \wedge x \neq y \end{array}\right] < \frac{1}{p(n)}$$

If in addition the following also hold, the collection is a collection of trapdoor pseudo-permutation

3. *Easy to find pre-image with trapdoor:* There is a trapdoor function t and polynomial time algorithm F^{-1} such that for every $(i, t(i))$ and for every $\mathcal{D}_i$, it holds that $F^{-1}(t, f_i(x)) = y$, and $f_i(x) = f_i(y)$.

Note that in the third condition of last definition, we will not force $x = y$. It is reasonable because permutation is pseudo-permutation, and it is hard to find two different pre-images for the same image. Please also note that F^{-1} is tend to find one of the pre-images for $f_i(x)$, rather than to find x.

A function as a member of a collection of pseudo-permutations is called a *pseudo-permutation*. A pair (x, y) is a *collapse* of f if $f(x) = f(y)$, and $x \neq y$. A collection of **claw-free pairs of pseudo-permutations** is a collection of claw-free pairs of functions in which each function is a pseudo-permutation.

Definition A.5 [Collection of Chameleon Hash Functions] A collection of chameleon hash functions is a collection of hash functions $S^m = \{ f_i \in I \mid f_i \colon \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^m \}$ that satisfies following conditions:

1. ***Easy sample and compute:*** There exist a generating algorithm $G \in PPT$, a sampling algorithm S, and a evaluation algorithm $E \in PPT$, so that $G[1^n] \in \{0,1\}^n \cap I$. $\forall i, S[i]$ is the uniform distribution on $\{0,1\}^*$. For all $x \in \{0,1\}^*, E[i,x] = f_i(x)$.
2. ***Collision resistance:*** For all collision finding algorithm $A \in PPT$, all polynomial p, for sufficiently large n,

$$\mathtt{Pr}\begin{bmatrix} i \leftarrow G[1^n], ((m_1, r_1), (m_2, r_2)) \leftarrow A[i], \\ m_1 \neq m_2 : \ f_i(m_1, r_1) = f_i(m_2, r_2) \end{bmatrix} < \frac{1}{p(n)}$$

3. ***Easy to make collision with trapdoor:*** There exists a trapdoor function t and a polynomial time algorithm TF such that $\forall i \in I$, and trapdoor $(i, t(i))$, it holds that for any (m_1, r_1, m_2), $TF(i, t(i), m_1, r_1, m_2) = r_2$, and $f_i(m_1, r_1) = f_i(m_2, r_2)$.
4. ***Uniformity:*** For all $i \in I$, $\forall m \in \{0,1\}^*$, the random variable $f_i(m, r)$ is uniformly distributed on $\{0,1\}^m$ whenever $r \in I$ uniformly chosen.

Applied Public Key Infrastructure
J. Zhou et al. (Eds.)
IOS Press, 2005

Author Index