

TRUSTED INFORMATION

The New Decade Challenge

Edited by
Michel Dupuy
Pierre Paradinas



IFIP



**KLUWER
ACADEMIC
PUBLISHERS**

TRUSTED INFORMATION
The New Decade Challenge

IFIP - The International Federation for Information Processing

IFIP was founded in 1960 under the auspices of UNESCO, following the First World Computer Congress held in Paris the previous year. An umbrella organization for societies working in information processing, IFIP's aim is two-fold: to support information processing within its member countries and to encourage technology transfer to developing nations. As its mission statement clearly states,

IFIP's mission is to be the leading, truly international, apolitical organization which encourages and assists in the development, exploitation and application of information technology for the benefit of all people.

IFIP is a non-profitmaking organization, run almost solely by 2500 volunteers. It operates through a number of technical committees, which organize events and publications. IFIP's events range from an international congress to local seminars, but the most important are:

- The IFIP World Computer Congress, held every second year;
- open conferences;
- working conferences.

The flagship event is the IFIP World Computer Congress, at which both invited and contributed papers are presented. Contributed papers are rigorously refereed and the rejection rate is high.

As with the Congress, participation in the open conferences is open to all and papers may be invited or submitted. Again, submitted papers are stringently refereed.

The working conferences are structured differently. They are usually run by a working group and attendance is small and by invitation only. Their purpose is to create an atmosphere conducive to innovation and development. Refereeing is less rigorous and papers are subjected to extensive group discussion.

Publications arising from IFIP events vary. The papers presented at the IFIP World Computer Congress and at open conferences are published as conference proceedings, while the results of the working conferences are often published as collections of selected and edited papers.

Any national society whose primary activity is in information may apply to become a full member of IFIP, although full membership is restricted to one society per country. Full members are entitled to vote at the annual General Assembly, National societies preferring a less committed involvement may apply for associate or corresponding membership. Associate members enjoy the same benefits as full members, but without voting rights. Corresponding members are not represented in IFIP bodies. Affiliated membership is open to non-national societies, and individual and honorary membership schemes are also offered.

TRUSTED INFORMATION

The New Decade Challenge

*IFIP TC11 16th International Conference on
Information Security (IFIP/Sec'01)
June 11-13, 2001, Paris, France*

Edited by

Michel Dupuy
*SGDN/DCSSI/CERTA
France*

Pierre Paradinas
*Gemplus Labs
France*

KLUWER ACADEMIC PUBLISHERS
NEW YORK, BOSTON, DORDRECHT, LONDON, MOSCOW

eBook ISBN 0-306-46998-7

Print ISBN 0-792-37389-8

©2002 Kluwer Academic / Plenum Publishers, New York

233 Spring Street, New York, N. Y. 10013

Print ©2001 Kluwer Academic Publishers, Boston

All rights reserved

No part of this eBook may be reproduced or transmitted in any form or by any means, electronic, mechanical, recording, or otherwise, without written consent from the Publisher

Created in the United States of America

Visit Kluwer Online at:

<http://www.kluweronline.com>

and Kluwer's eBookstore at:

<http://www.ebooks.kluweronline.com>

Contents

Preface	ix
IFIP/Sec'01 Conference Committees	xi

PART ONE eSociety

1. PyTHIA: Towards Anonymity in Authentication <i>D. GRITZALIS, K. MOULINOS, J. ILIADIS, C. LAMBRINOUDAKIS and S. XARHOULACOS</i>	1
2. Certificate Based PKI and B2B E-Commerce: Suitable Match or Not? <i>K.M. ANG and W.J. CAELLI</i>	19
3. Internet Anonymity: Problems and Solutions <i>C. ECKERT and A. PIRCHER</i>	35

PART TWO TTP Management and PKI

4. Reducing Certificate Revocation Cost using NPKI <i>A. LEVI and Ç. KAYA KOÇ</i>	51
5. The Need and Practice of User Authentication and TTP Services in Distributed Health Information Systems <i>B. BLOBEL and P. PHAROW</i>	61

PART THREE Smart Card

6. Is the Performance of Smart Card Cryptographic Functions the Real Bottleneck? <i>K. MARKANTONAKIS</i>	77
7. Modelling Audit Security for Smart-Card Payment Schemes with UML-SEC <i>J. JÜRJENS</i>	93

PART FOUR Security Protocols

8. Strong Forward Security <i>M. BURMESTER, V. CHRISSIKOPOULOS, P. KOTZANIKOLAOU and E. MAGKOS</i>	109
9. Secret Sharing and Visual Cryptography Schemes <i>A. DE BONIS and A. DE SANTIS</i>	123
10. A Two-level Time-Stamping System <i>A. GABILLON and J. BYUN</i>	139
11. Security Analysis of the Cliques Protocols Suites: First Results <i>O. PEREIRA and J-J. QUISQUATER</i>	151

PART FIVE Secure Workflow Environment

12. A Secure Workflow System for Dynamic Collaboration
J. S. PARK, M. H. KANG and J. N. FROSCHER167
13. On Securely Scheduling a Meeting
*T. HERLEA, J. CLAESSENS, B. PRENEEL, G. NEVEN, F. PIESSENS
and B. De DECKER*183
14. Modeling and Analyzing Separation of Duties in Workflow Environments
K. KNORR and H. STORMER199

PART SIX Secure Group Communications

15. Group Security Association (GSA) Management in IP Multicast
T. HARDJONO, M. BAUGHER and H. HARNEY 213
16. Communication-Efficient Group Key Agreement
Y. KIM, A. PERRIG and G. TSUDIK 229

PART SEVEN Security Policies

17. Going Beyond MAC and DAC Using Mobile Policies
A. FAYAD, S. JAJODIA, D. FAATZ and V. DOSHI 245
18. An Access Control Model for Data Archives
*P. BONATTI, E. DAMIANI, S. De CAPITANI Di VIMERCATI and
P. SAMARATI* 261

PART EIGHT Risk Management

19. Checklist-Based Risk Analysis with Evidential Reasoning
S. CHO and Z. CIECHANOWICZ277
20. Improving the Protection of Assets in Open Distributed Systems by
Use of X-ifying Risk Analysis
A. FRISINGER 293
21. The Security Model to Combine the Corporate and Information Security
T. VIRTANEN 305
22. Design Criteria to Classified Information Systems Numerically
T. VIRTANEN317

PART NINE Network Security and Intrusion Detection

23. Deception: A Tool and Curse for Security Management
M. WARREN and W. HUTCHINSON327
24. A Methodology to Detect Temporal Regularities in User Behavior for
Anomaly Detection
A. SELEZNYOV 339
25. ADeLe: An Attack Description Language for Knowledge-Based
Intrusion Detection
C. MICHEL and L. MÉ353
26. Sleepy Watermark Tracing: An Active Network-Based Intrusion
Response Framework
X. WANG, D. S. REEVES, S. F. WU and J. YUILL369

PART TEN Trusted Platforms

27. An Efficient Software Protection Scheme
A. MAÑA and E. PIMENTEL 385
28. Protecting the Creation of Digital Signatures with Trusted Computing
 Platform Technology Against Attacks by Trojan Horse Programs
A. SPALKA, A.B. CREMERS and H. LANGWEG 403

PART ELEVEN Trusted System Design and Management

29. Security Concerns for Contemporary Development Practices:
 A Case Study
T. TRYFONAS and E. KIOUNTOUZIS 421
30. A Paradigmatic Analysis of Conventional Approaches for Developing
 and Managing Secure IS
M. T. SIPONEN 437
31. Redefining Information Systems Security: Viable Information Systems
M. KARYDA, S. KOKOLAKIS and, E. KIOUNTOUZIS 453
32. Extended Description Techniques for Security Engineering
G. WIMMEL and A. WISSPEINTNER 469

This Page Intentionally Left Blank

Preface

Since the early eighties IFIP/Sec has been an important rendezvous for Information Technology researchers and specialists involved in all aspects of IT security. The explosive growth of the Web is now faced with the formidable challenge of providing trusted information.

IFIP/Sec'01 is the first of this decade (and century) and it will be devoted to "Trusted Information - the New Decade Challenge"

This proceedings are divided in eleven parts related to the conference program. Sessions are dedicated to technologies: *Security Protocols, Smart Card, Network Security and Intrusion Detection, Trusted Platforms*. Others sessions are devoted to application like *eSociety, TTP Management and PKI, Secure Workflow Environment, Secure Group Communications*, and on the deployment of applications: *Risk Management, Security Policies and Trusted System Design and Management*.

The year 2001 is a double anniversary. First, fifteen years ago, the first IFIP/Sec was held in France (IFIP/Sec'86, Monte-Carlo) and 2001 is also the anniversary of smart card technology. Smart cards emerged some twenty years ago as an innovation and have now become pervasive information devices used for highly distributed secure applications. These cards let millions of people carry a highly secure device that can represent them on a variety of networks.

To conclude, we hope that the rich "menu" of conference papers for this IFIP/Sec conference will provide valuable insights and encourage specialists to pursue their work in trusted information.

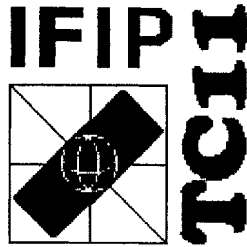
Michel DUPUY
General Chair

Pierre PARADINAS
Program Chairman

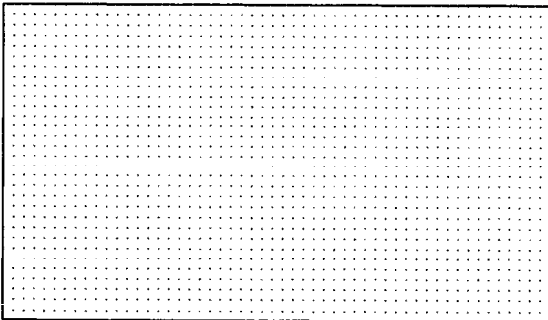
This Page Intentionally Left Blank

IFIP/Sec'01
TRUSTED INFORMATION
The New Decade Challenge

is organised by



and sponsored by



This Page Intentionally Left Blank

IFIP/Sec'01 Conference Committees

Conference Chairs

General Chair: Michel DUPUY (SGDN/DCSSI/CERTA, France)
Program Chairman: Pierre PARADINAS (Gemplus Labs, France)

Program Committee

William J CAELLI (Queensland University of Technology, Australia)
Bart De DECKER (K.U. Leuven, Belgium)
Yves DESWARTE (LAAS-CNRS, France)
Michel DUPUY (SGDN/DCSSI/CERTA, France)
Jan ELOFF (Rand Afrikaans University, South Africa)
Geoff FAIRALL (Zimbabwe)
Pierre GIRARD (Gemplus, France)
Dimitris GRITZALIS (Athens Technical Education Institute, Greece)
Pieter HARTEL (United Kingdom)
Sokratis K. KATSIKAS (University of the Aegean, Greece)
Juha E. MIETTINEN, (Sonera Corporation, Finland)
Mongi MILED (Centre National de l'Informatique, Tunisia)
György PAPP (V.R.A.M. Commination, Hungary)
Pierre PARADINAS (Gemplus Labs, France)
Hartmut POHL (Fachhochschule Bonn-Rhein-Sieg, Germany)
Reinhard POSCH (IAIK-Graz University of Technology, Austria)
Jean-Jacques QUISQUATER (U.C. Louvain, Belgium)
Kai RANNENBERG (Microsoft Research Cambridge, United Kingdom)
Pierangela SAMARATI (Universita' di Milano, Italy)
S. H. von SOLMS (Rand Afrikaans University, South Africa)
R. von SOLMS (Port Elizabeth Technikon, South Africa)
Jacques STERN (École Normale Supérieure, France)
Leon A.M. STROUS (De Nederlandsche Bank, The Netherlands)
Jozef VYSKOC (VaF, s.r.o., Slovak Republic)
Willis H. WARE (The Rand Corporation, USA)
Lam Kwok YAN (PrivyLink Pte, Singapore)
Louise YNGSTROM (University of Stockholm, Sweden)

Referees

Olivier BAUDRON (École Normale Supérieure, France)
 Jan. Van Den BERGH (K.U.Leuven, Belgium)
 Olivier CASTAN (SGDN/DCSSI/CERTA, France)
 Lenka FIBIKOVA (University of Essen, Germany)
 Adam FIELD (University of Twente, The Netherlands)
 Pierre-Alain FOUQUE (École Normale Supérieure, France)
 Matthieu GRALL (SGDN/DCSSI, France)
 Stefanos GRITZALIS (University of the Aegean, Greece)
 Helena HANDSCHUH (Gemplus, France)
 Jaap-Henk HOEPMAN (University of Twente, The Netherlands)
 Pierre JANSEN (University of Twente, The Netherlands)
 Guy-Armand KAMENDJE (IAIK-Graz University of Technology, Austria)
 Spyros KOKOLAKIS (University of the Aegean, Greece)
 Herbert LEITOLD (IAIK-Graz University of Technology, Austria)
 Peter LIPP (IAIK-Graz University of Technology, Austria)
 Gregory NEVEN (K.U.Leuven, Belgium)
 Vincent NICOMETTE (LAAS/CNRS, France)
 Elisabeth OSWALD (IAIK-Graz University of Technology, Austria)
 Pascal PAILLIER (Gemplus, France)
 Udo PAYER (IAIK-Graz University of Technology, Austria)
 Beatrice PEIRANI (Gemplus, France)
 Fabien A.P. PETICOLAS (Microsoft Research Cambridge, United Kingdom)
 Franck PIESSENS (K.U.Leuven, Belgium)
 David POINTCHEVAL (École Normale Supérieure-CNRS, France)
 Thomas PORNIN (École Normale Supérieure, France)
 Martin STANEK (Comenius University, Slovak Republic)
 Andreas STERBENZ (IAIK-Graz University of Technology, Austria)
 Vassilis TSOUMAS (University of the Aegean, Greece)
 Bart VANHAUTE (K.U.Leuven, Belgium)

Organizing Committee

Sylvie LEKIEFFRE (SEE, France)
 Catherine VANNIER (SEE, France)

PyTHIA: Towards Anonymity in Authentication

Dimitris GRITZALIS¹, Kostantinos MOULINOS¹, John ILIADIS²,
Costas LAMBRINOUDAKIS², Steven XARHOULACOS²

¹ *Dept. of Informatics, Athens University of Economics and Business
76 Patission St., Athens GR-10434, Greece, e-mail: {dgrit,kdm}@aueb.gr*

² *Dept. of Information and Communication Systems, University of the Aegean
30 Voulgaroktonou St., Athens GR-11472, Greece, e-mail: {jiliad,clam,stx}@aegean.gr*

Abstract There is a scale between authentication and anonymity, which is currently leaning towards the side of authentication, when it comes to e-commerce. Service providers and merchants are usually keeping track of user-related information in order to construct behavioural profiles of their customers. Service providers and merchants also correlate profiles of this kind, stemming from different sources, in order to increase their profit. This correlation is usually performed with the use of Unified Codes. Authentication, confidentiality, integrity, authentication, and non-repudiation are necessary functionalities for enabling e-commerce. Most of the currently used mechanisms that support these services do not provide anonymity. This paper presents PyTHIA, a mechanism, which is based on the use of Message Digest Algorithms and the intermediation of Trusted Third Parties in order to provide anonymity to e-commerce users who have to authenticate themselves in order to access services or buy goods from service providers and merchants respectively. With PyTHIA e-commerce users are able to authenticate without giving away any personal data and without using Unified Codes. In addition, PyTHIA ensures that service providers and merchants can effectively trace a customer in case he behaves maliciously.

Keywords Privacy Enhanced Technologies (PET), Security, Privacy, Anonymity, Certificates, Trusted Third Party, (TTP), PyTHIA

1. INTRODUCTION

Electronic Commerce (e-commerce) is expected to dominate business transactions in the future. Virtual markets and trade conducted over the Internet are anticipated to grow at an explosive rate. In 1996, Amazon.com recorded sales of less than \$16 million, while in 1997 it sold \$148 million worth of books [IMR98]. E-commerce eliminates the need of intermediaries, minimizes the product cost, and provides customers with worldwide market access. These are due to the wide use of data network technologies, and the evolution of the World Wide Web (Web). The Web attracted the average user to electronic business with its user-friendly interface. Despite its security problems [GRI99], the Web enabled people to interact using multimedia content.

In order to promote their sales, merchants are establishing new ways of collecting, processing and exchanging user data. Advertising is increasingly shifting towards the Web, as this communication channel fulfils promises for better targeting, more efficient response and more accurate audience measurement. During 1996, Internet advertising increased by a factor of ten from \$20 million to \$200 million while during 1997 it has risen to \$600 million. The year 2000, \$40 billion expected to be spent on Internet advertising [IMR98].

In order to measure the audience's marketing preferences and customize their product lines to specific user needs, merchants collect online personal data when a customer connects to their site. They further use advanced scientific techniques, such as data mining, to compile and analyse the data they had already collected, to form profiling databases. A user profile is a collection of personal data that uniquely identifies a person. The data collected for e-commerce purposes become critical tools in tracing potential clients' consuming patterns.

The collection and processing of personal data may lead to private and family life violation, thus discouraging the public from using new technologies. According to a Business Week/Harris poll [BUS98], lack of privacy in communications is the main reason of being off the Internet for the great majority of potential users. Users consider the lack of privacy to be a deterrent against e-commerce, even more than cost, difficulties in use and unwanted marketing messages. This situation would have a profound impact on the growth of the Internet with further consequences on the evolution of e-commerce and increase of advertising revenues [BUS98].

The antidote to online privacy infringement consists of channels that do not reveal the identity of the communicating parties. Such channels are called *anonymous channels*. Internet operation should be based on the principle of anonymity. If individuals wish to maintain the level of privacy

they enjoy in real world, they should be given the choice for anonymity in the Internet.

Deploying e-commerce infrastructures requires among others entity authentication, and confidentiality and integrity of the transmitted data. Protecting the confidentiality and integrity of data does not usually degrade the levels of privacy. Authentication, however, contrasts with anonymity. There is a scale between these two, and it is leaning towards the side of authentication when it comes for e-commerce. This is due to the fact, that strong authentication is based on the disclosure of the identity of the involved parties. On the contrary, anonymous communications do not reveal the identity of the involved parties. As a result, new technologies should evolve permitting the authentication of users while also facilitating their anonymity.

This paper presents an authentication mechanism that requires the intermediation of Trusted Third Parties (TTP), enabling Web users to authenticate themselves against the sites they visit and at the same time refrain from revealing any personal information. The mechanism averts personal data profiling and enables companies to trace the identity of a customer in case of fraud.

The paper is organized as follows: In section 2 an overview of Privacy Enhancing Technologies is presented, while in section 3 a framework is presented, the privacy mechanism should operate within. In section 4 we analyse the operation of this mechanism. Section 5 contains a discussion on the inner-workings of the mechanism and ideas for future enhancements. Finally, in section 6 some concluding remarks are provided.

2. OVERVIEW OF PRIVACY ENHANCING TECHNOLOGIES

Privacy Enhancing Technologies (PET) include those technologies developed to protect users from revealing their identity when they communicate with each other. In this section we focus on PET applied to Internet technologies.

The various PET mechanisms are strongly interrelated; many of them are based on recent technological developments and some blur the traditional distinctions between setting, implementing and enforcing privacy guidelines. The various mechanisms for the protection of privacy on global networks, according to their purpose, can be categorized as follows [OEC99].

2.1 Minimizing disclosure and collection of personal data

This category includes the following mechanisms:

- Management of cookies. Cookies comprise text files, formulated during the connection of a Web browser to a Web server via HTTP, and enabling the Web server to trace the on-line behaviour of the client.
- Anonymous re-mailers are e-mail servers permitting users to send electronic messages without revealing their identity.
- Anonymous re-webbers are proxies providing users with the ability to anonymously visit web sites.
- Anonymous payment systems. The most anonymous means of digital payment is electronic cash. Electronic cash comprises an electronic payment system that protects user anonymity and payment untraceability. In general, electronic cash schemes achieve these goals via digital signatures [LAW96].
- Digital certificates are digital tokens, issued by TTP, confirming the identity of the token holder. Digital certificates typically carry personal information. There is one category of certificates, which are used to confirm that a particular user is authorized to make a specific kind of transaction. These mechanisms do not directly reveal personal information.
- Anonymous profiles are those, which do not contain the personal identification information of a user. Each user is assigned a numerical identifier using cookies.

2.2 Informing users about on-line privacy policies

Various ways exist in order to inform users about the privacy policies adopted by web sites, including posted privacy policies, terms and conditions, and digital labels. Infrastructures exist supporting this practice. The most popular include TRUSTe, BBOnLine, the OECD Privacy Generator, and P3P. The latter is a specification, developed by W3C [W3C99], enabling Web sites to express privacy policies in a standard format.

2.3 Providing users with options for personal data disclosure and use

Three practices belong to this category:

1. *On-line negotiation* of privacy standards through digital labels.

2. *Opting-in*, which refers to optional data fields and click-box choices commonly used by several Web sites to mark as optional several fields on the forms they use to collect personal data.
3. *Opting-out*, which refers to the ability of users to control the use of personal data they possess, either previously made known, or those being publicly available. This category includes the following mechanisms:
 - a) Controlling the use of personal data following the completion of collection, which refers to a common practice of several Web sites giving users the choice to change their mind and withdraw their consent to collect personal data. This is usually accomplished via e-mail.
 - b) Preventing the receipt of unsolicited e-mail advertising. The most popular mechanism of this category is Robinson lists, which include the names of all those people not wishing to receive electronic messages of advertising content. Legal authorities such as the national Data Protection Authorities in Europe usually dispatch the Robinson Lists to the public.
 - c) Opting-out of anonymous profiling which refer to the ability of users to erase collected personal data.

2.4 Providing access to personal data

This category includes off-line or on-line mechanisms permitting users to access personal data they have previously release. The Open Profiling Standard (OPS) is a standard for exchanging information between individuals and service providing parties. In addition, OPS supports user privacy by giving the end-user the ability to control the release of their personal data and track their exchange and usage [OEC98]. The standard specifies the following [W3C97]:

- Naming issues and rights of authorities regarding profile data.
- Varying levels of security of communicated data.
- Elementary profile operations such as *profile read* and *profile write*.

2.5 Protecting privacy through trans-border data flow contracts

This category includes all legal agreements and contracts between different countries, with respect to the protection of personal data. When studying these agreements, particular attention should be paid to the characteristics of data flow, including the nature of the data, the purpose and duration of the processing, the country of origin and destination of the flow, the data protection laws in the involved countries, and the security measures taken.

In addition, identifying the protection level “adequacy” offered by the destination country has become the most distinct debate with regard to trans-border data flow. The European Union Directive 95/46 [EUR95] and the Council of Europe Model Contract of 1992 [OEC99] have adopted the term “adequate level of protection”, while OECD Guidelines state that trans-border flows may be restricted in case that no “equivalent” protection exists [EUR95].

Furthermore, one should define what the “adequate” level of protection is. For this reason, the European Union has set up a Working Party (under Articles 29 and 31 of the Directive) [EUR95]. Among other duties, this Working Party is responsible for giving the Commission an opinion on the level of data protection in the European Union Member States, as well as in third countries. In case there is no national legal framework, other means may be utilized in order to identify the adequacy of the data protection level. For example, the U.S. Federal Trade Commission follows a system of self-regulation, which established a set of data protection principles, called *Safe Harbour*. United States companies reassure their European customers that they respect individual privacy by compiling a list of companies complying with Safe Harbour principles.

2.6 Enforcing Privacy Principles

Enforcing privacy principles can be distinguished in two categories [OEC99]:

1. *Ensuring compliance with privacy standards.* Companies follow this proactive approach by reassuring their customers with regard to their compliance with national and international data protection practices and laws. In essence, data protection auditing is performed either by external or internal entities, which confirm that the examined organization actually has activated procedures and has taken measures to protect personal data. The entities that perform the audit can be internal data protection officers, third party reviewers, standards organizations, accounting firms, industrial firms, etc.
2. *Complaint resolution procedures for breaches of privacy standards.* Individuals follow this reactive approach when they believe that their personal lives have been violated. The resolution is usually made between the data subject concerning the breach and the data controller. Other means of resolution include private sector and industry bodies certification schemes, and administrative, civil and criminal proceedings.

2.7 Educating users and the private sector

Except for the entities directly involved in data protection matters, ISPs, Service Providers, and companies should promote the education of users with respect to mechanisms and practices they can use to protect their personal data.

There are, currently, several organizations that undertake this educative task, including Project OPEN (the Online Public Education Network), the U.S. Direct Marketing Association, the Centre For Democracy and Technology, the Electronic Privacy Information Centre “Call for Action” and TRUSTe, among others.

3. TOWARDS ANONYMITY IN AUTHENTICATION

Anonymous authentication is expected to contribute in the growth of e-commerce. However, there is a reverse analogous relationship between anonymity and authentication. E-commerce involves the use of on-line services and real time communication. The latter adds new challenges in protecting user anonymity while requiring the authenticated presence of users. We present a mechanism, called PyTHIA, which supports anonymous and authenticated communications. The three axes, our mechanism is based on, are the following:

1. Communication and user *anonymity* as a means to support anonymous profiling.
2. The existing *legal framework with regard to personal data protection*, which influences the deployment and release of anonymous communication.
3. *Authentication* in wide area networks, which is effectively implemented by using TTP services.

3.1 Anonymity

Anonymity is examined as a service offered and ensured by communication networks. Anonymous communication is a powerful means individuals have to ensure their privacy. One can distinguish four types of communication where the sender's physical identity is partly hidden [FRO96]:

1. *Traceable anonymity*, giving no clue about the sender's identity and leaving this information in the hands of an intermediary. Typically, the sender should trust the intermediary. Although traceable anonymity offers

the lowest security it permits the recipient of a message to trace back the identity of the sender in cases of repudiation between the involved parties.

2. *Untraceable anonymity* in which there is no way of revealing the identity of the sender.
3. *Traceable pseudonymity*, which assigns a pseudonymous (or 'nym') to the sender of message. The pseudonymous can be used to trace the real identity of the sender.
4. *Untraceable pseudonymity*, where a pseudonymous is assigned to the sender of the message as in traceable pseudonymity. However, this cannot be used in order to trace the real identity of the user.

Anonymity has both beneficial and harmful implications in peoples' lives. For the purpose of this paper, we focus on

- a) privacy protection as a means for enabling anonymous profiling,
- b) avoiding impersonation,
- c) avoiding fraud in on-line transactions.

3.2 Legal framework concerning data protection

Although profiling may not change the amount of actual collected data concerning a person, organizing the data into searchable form reduces the person's privacy by permitting correlations that were previously impossible. In order to limit the impact of such processing on individuals' personal lives, several data protection laws have been enacted worldwide. The most renown is the European Directive 95/46, "*On the protection of individuals with regard to the processing of personal data and on the free movement of such data*" [EUR95], which sets the prerequisites for data owners and processors for collecting, processing and exchanging personal data. The U.S. government promotes the notion of "self regulation", a set of data protection rules applying to a plurality of market sectors, the content of which has been primarily determined by members of the specific trade sector.

Special emphasis has been placed on the use of Unified Codes, in several interpretations of 95/46 Directive. For example, article 8 of the Greek National Data Protection Law (L. 2472/97) [DAT97], states that the use of Unified Codes as a means of cross-linking personal data files, belonging to different data controllers, should be prohibited. This is due to the fact that using Unified codes may result in forming personal profiles within wider communities.

3.3 Trusted Third Parties

Not all TTP services can be supported only by technological means (e.g. in the case of non-repudiation service, there should be a legal body that

recognizes digital signatures as legal evidence). In addition, functions supported by technology may sometimes fail due to errors. To cover inadequacies presented in all these cases, entities using a PKI need to be aware of the legal principles and frameworks that support their use of PKI facilities and TTP mechanisms.

4. PYTHIA

We present a prototype for a mechanism called PyTHIA (PrivacY Through Hashes In Authentication) that supports *traceable anonymity*. PyTHIA users own a cryptographic construct called Privacy-Protected Authentication Token (PPAT), issued by an appropriate authority.

We consider Trusted Third Parties (TTP) can undertake this role, in the form of a value-added service. PPAT owners can authenticate themselves against Web sites offering products or services, using this token. However, no element of their identity is disclosed. If a user later repudiates his actions, the TTP can help in adjudicating the dispute by revealing the true identity of the entity, which used a specific PPAT to authenticate itself against a site.

The mechanism uses the security infrastructure provided by TTPs and digital certificates, as a means to trace the — certified — identity of users whenever this is needed. PyTHIA users must have obtained a digital certificate from a TTP, before requesting a PPAT and using PyTHIA. Although we were considering X.509v3 certificates [ISO95] while developing the mechanism, PyTHIA can make use of other categories kinds of certificates as well.

Throughout the presentation of PyTHIA, we assume that Alice wishes to use the mechanism to protect her privacy, while authenticating herself at Bob's web site. We also assume that Alice already possesses a valid certificate $Cert_A$ from a TTP called Trent, before requesting a PPAT from that TTP.

4.1 PPAT generation

We present the basic elements a PPAT comprises of, before analysing the PPAT generation process. The first element is the output of a collision-free hash function. The input to this function must be $Cert_A$ and a pseudorandom value RV produced at the time of PPAT generation. Actually, the first element of the PPAT is the output of the hash function applied n times to the aforementioned data. Trent chooses n , and the reason behind this choice is explained in the next section where we present in detail the PPAT generation process.

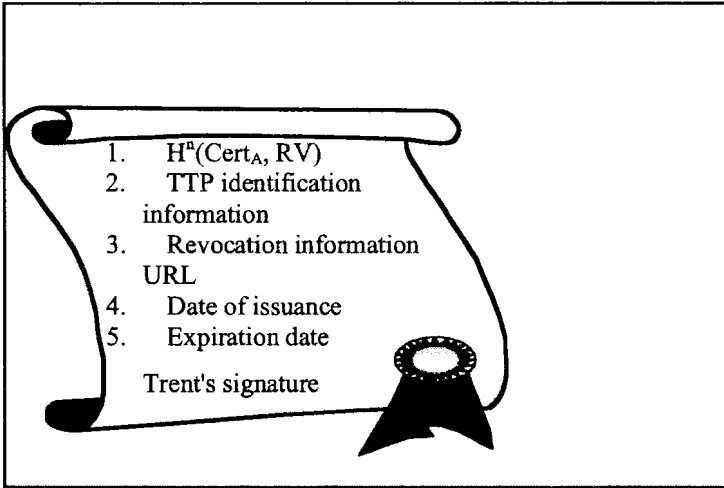


Figure 1: Privacy-Protected Authentication Token

The second element of the PPAT is identification information of Trent. The third element of the PPAT is a Uniform Resource Locator [LEE98] pointing to Trent's PPAT revocation status service. PPATs get revoked when the respective user certificates are revoked. The fourth and the fifth element refer to the date and time of issuance of the PPAT, as well as its expiration date and time. This must be equal to the expiration date of the respective digital certificate Alice has obtained from Trent.

Alice initiates the PPAT generation process by requesting a PPAT from Trent. Trent requests from Alice to authenticate herself using the certificate Trent has issued for Alice at a previous time. Trent computes the time period between the expiration date of Cert_A and the current date. Trent proceeds with expressing the aforementioned time period in a predefined time unit (for the sake of simplicity we will be using *hours* as a specific time unit for our example). Having computed the amount n of hours contained in the aforementioned time period, Trent computes $H^n(\text{Cert}_A, RV)$.

Finally, Trent gathers the output of the aforementioned hash function, the information contained in the second and third field of the PPAT, the current time (fourth field) and the expiration date of Cert_A (fifth PPAT field) and digitally signs them, using a private key reserved for that purpose only. The resulting construct is the PPAT of Alice (PPAT_A).

Trent stores PPAT_A in his protected database, along with a link to (or a copy of) Cert_A , enabling him to quickly identify the owner of PPAT_A , whenever this is needed. Trent communicates to Alice $H(\text{Cert}_A, RV)$, that is the output of the hash function applied once on Cert_A and RV . Trent also communicates to Alice the PPAT_A itself, the number n and the RV . Alice stores this information at her protected, local repository.

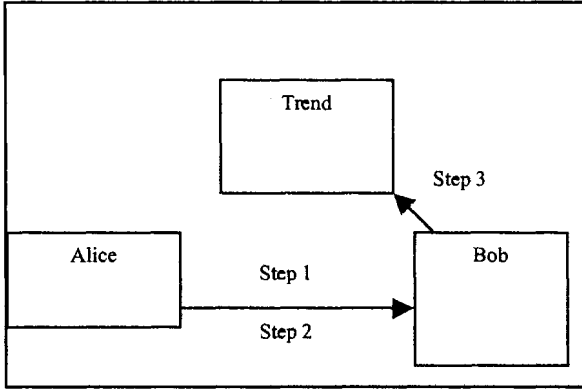


Figure 2: PPAT Authentication

4.2 Using PPAT to authenticate

Alice visits Bob and performs an action, which requires Alice's authentication lest she repudiates this action at a later stage. Alice communicates to Bob (Step 1) the $PPAT_A$. Although $PPAT_A$ does not disclose any personal information of Alice, it identifies Alice as a specific entity, carrying this unique identification badge and registered with the TTP that issued the $PPAT_A$. Alice must proceed with calculating the amount of hour k that has passed since the time the $PPAT_A$ was issued. Alice sends (Step 2) to Bob H^{n-k} , by recursively applying the hash function H $n-k-1$ times to the value $H(\text{Cert}^A, RV)$. Bob calculates k as well and verifies that the first element of the received $PPAT_A$ derives by applying k times the hash function H to the value H^{n-k} he has received from Alice. Alice is authenticated, since only Alice (and the TTP) could produce H^{n-k} at that time.

Finally, Bob has to send his identity (Step 3), $PPAT_A$, and H^{n-k} to Trent *or have this information time-stamped by an independent Time-stamping Authority (TSA)*.

If Alice repudiates her actions at a later stage, Bob communicates the aforementioned timestamp to Trent, or requests from Trent to search his protected repository and locate the information Bob had sent him at the time Alice visited Bob. Since the exact time this information was made available to Bob could be verified and this information could be produced at that time only by Alice, therefore Alice cannot repudiate having visited Bob then.

However, Alice could claim having performed different actions at Bob's site, at that time. Bob has no means to prove that Alice had performed indeed the actions he claims she had. We discuss possible extensions to the mechanism to support this, in later sections.

4.3 PPAT revocation

Bob can verify the revocation status of $PPAT_A$, by querying the appropriate TTP service (the URL for this service is the third element in $PPAT_A$). Bob must send to this service the $PPAT_A$ and the service will check the status of $Cert_A$ and return that to Bob. The status of $PPAT_A$ always depends on the status of $Cert_A$.

5. DISCUSSION

PyTHIA is an authentication mechanism that proactively protects the privacy of personal data belonging to the authenticating entities. PyTHIA does not address privacy issues related to the underlying communication protocols and mechanisms used at a transaction, like the mechanisms presented in section 2.1 do. However, PyTHIA could be used in conjunction with some of those mechanisms, in order to decrease the leak of personal data due to the underlying communication mechanisms.

PyTHIA users do not need to trust that the entities they communicate with (and authenticate against) shall not attempt to collect their personal data, or that they follow any specific policy regarding privacy. The privacy mechanisms presented in sections 2.2 and 2.3 depend on that kind of trust, and primarily on the trust, users place on the authorities that audit the privacy policy - and its implementation throughout the business functions - of businesses.

Furthermore, PyTHIA users do not need to control the amount of personal data they give away, nor do they need to use mechanisms to retract personal data they had given away at a previous time. PyTHIA does not release any personal data at all, therefore it should not be required to provide mechanisms for data subjects to access the personal data 2.4 a company has collected for them.

PyTHIA could release, indirectly, personal data. In detail, personal data could be released through inter-business data mining. Future work on PyTHIA may provide solutions to this problem, as well. However, preventing inter-business data mining can also be achieved by using PyTHIA only in environments where privacy regulatory frameworks (as those described in section 2.5) and voluntary compliance schemes (as those described in sections 2.2 and 2.6) apply. The technical measures by themselves could prove to be inadequate, either due to misuse from the data subjects themselves, or due to deliberate attacks by entities that attempt to violate the privacy of the aforementioned data subjects.

Technical measures should be enforced with related regulatory frameworks, and wide dissemination of information both on the technical measures and on the legal frameworks towards users. User awareness on privacy matters should be encouraged by authorities who regulate the protection of personal data, and should be promoted by entities that can successfully push information to end-users, such as ISPs, renown companies and organizations targeted to informing the public on privacy matters (also see section 2.7).

PyTHIA does not provide a mechanism for protecting the confidentiality or the integrity either of the exchanged transactional information, or of the exchanged information concerning the mechanism itself. Other mechanisms (e.g. SSL [FRE96] without client-side authentication) could be used in order to protect the confidentiality and integrity of information exchanged between Alice, Bob and Trent.

While investigating PyTHIA we have come up with various ways for providing Alice with the necessary information to authenticate herself against Bob. We have seen that the use of public key encryption could facilitate this task, in certain ways. However, we opted out of using public key encryption and we chose to use hash functions only, for a specific reason. If public key encryption was used, then in some scenarios a private key compromise would potentially reveal Alice's personal information to all the Web sites she had visited up to that time. Since personal data can be considered highly sensitive or confidential, depending on the place and time of their use by Alice or data collectors, we preferred to opt out of using public key encryption.

6. FUTURE WORK

Alice is using the PPAT to identify herself to the Web sites she is visiting. The PPAT does not contain any personal data therefore no such data is leaked to these Web sites. However, if two or more Web sites collude into cross-referencing the PPAT they have collected from their visitors, then anonymous user profiles could be constructed. PyTHIA could be improved to deal with this threat. Alice could request and obtain more than one PPAT at a time from Trent, each one containing a different pseudorandom value RV. If Alice obtains r PPAT from Trent, then she will be able to visit at most r Web sites, excluding any possibility for those sites to cross-reference their visitor databases and construct a user profile on Alice. This presupposes that Alice will be using a different PPAT for each Web site she visits and that she will use no PPAT twice. However, this scenario can be quite unrealistic, since the number r of Web sites Alice visits (and to which she has to

authenticate herself) could be rather high. Issuing a high number of PPAT would result in high computational burden for Trent and high communication burden between Trent and Alice.

There is a balance between the level of privacy Alice wishes and the computational and communication burden this entails (see Figure 3). Furthermore, managing a high number of PPAT may become difficult for Alice, since she will have to track the use of her PPAT, in order to ensure that a specific PPAT is not used twice or at least is not used in too many Web sites.

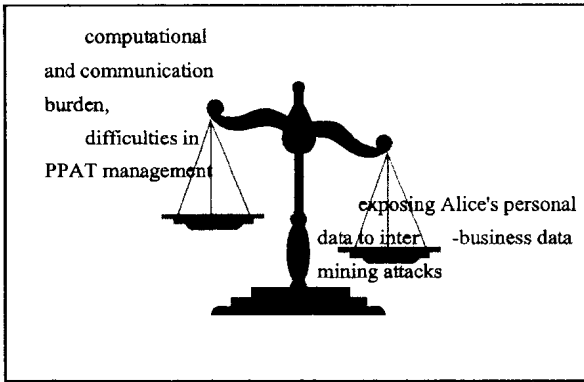


Figure 3: Consequences of managing numerous PPAT

Managing numerous PPAT could be facilitated if each PPAT is issued by Trent with a short, different - and potentially partially overlapping - validity period. Therefore, Alice must request a high enough number of PPAT in order to protect her privacy from inter-business data mining attacks, and at the same time minimize the consequences a very high number of PPAT requests would incur.

Another issue that has to be studied further in PyTHIA is to minimize the effects of a potential compromise of that part of the TTP that offers PyTHIA services. If Mallory succeeds in obtaining unauthorized access to the PyTHIA database, then Mallory would obtain personal data regarding all entities that have obtained PPAT from that TTP. All Mallory has to do is locate the PPAT of the entity, and retrieve the respective digital certificate. Trent could employ a mechanism to stall Mallory from discovering the aforementioned information and provide the time to deal reactively with the successful unauthorised access (block the access Mallory obtained to the database, or even monitor Mallory's activities and notify the PPAT whose identities have been revealed).

In order to stall Mallory, Trent could refrain from storing the PPAT themselves to the database, at PPAT generation time. Trent could store instead only the produced RVs in the database, and not link each RV to the corresponding PPAT and digital certificates.

$$H^n(\text{Cert}_i, \text{RV}_j), \forall i \in [1.. \text{NumberofIssuedCertificates}] \text{ and } \forall j \in [1.. \text{NumberofRandomValuesinDatabase}]$$

Equation 1: Mallory attempts to discover personal data for a PPAT owner, after having obtained unauthorized access to the PyTHIA database

This would increase much the time it would take for Mallory to discover the identity of a specific entity, since Mallory would have to retrieve the whole list of RVs, produce all the hashes described by Equation 1.

However, Trent would also have to perform all these computations whenever he would have to locate a specific digital certificate, based on a PPAT (e.g. when checking the revocation status of that PPAT). If the RV was stored in the PPAT, encrypted under Trent's private key, then Trent could immediately locate a digital certificate, based on the information provided by a PPAT, and at the same time if Mallory managed to obtain unauthorized access to the PyTHIA database, she would have to perform all the aforementioned computations.

Another improvement for PyTHIA concerns preventing Bob from claiming that Alice had visited him at an earlier time, than she really did. The present status of PyTHIA requires Bob to timestamp the authentication information he has received from Alice in order to prevent Bob from falsely claiming that Alice visited his site at an earlier point in time.

However, PyTHIA would be more efficient if Bob did not have to timestamp the aforementioned information. Solutions that would replace the need for Bob to communicate online with Trent or a TSA must be studied. We believe that these solutions could consist of including time-related information in the hashes produced by Alice, and making use of new technologies concerning digital signatures like forward-secure signatures [BEL99] or other cryptographic schemes.

7. CONCLUSIONS

We presented the prototype of a proactive mechanism for traceable anonymity. PyTHIA prevents any leak of personal data of a subject, when the subject is authenticated. PyTHIA can be used in conjunction with others, in order to provide a multilevel, integrated solution to the problem of privacy protection.

Furthermore, improvements to PyTHIA could prevent inter-business data mining, resulting in the construction of anonymous user profiles. There is still need for improvement in the suggested mechanism; the most important aspects that will be dealt with in the future are mentioned in section 6.

No PET mechanism by itself is sufficient for protecting privacy. Privacy clearly needs to be studied from a technical point of view. However, the technical mechanisms that protect privacy should be supported by an appropriate underlying legal infrastructure. Besides that, user awareness is a major issue. Until we achieve a satisfying degree of privacy-literacy, the privacy mechanisms and the legal infrastructures will not be able to operate efficiently.

REFERENCES

- [BEL99] M. Bellare, S. Miner, "A forward-secure digital signature scheme", *Lecture Notes in Computer Science*, Vol. 166, M. Wiener (Ed.), Springer-Verlag, 1999.
- [BUS98] Business Week, "A little net privacy please", 16 March 1998 (available at <http://www.businessweek.com>)
- [DAT97] *Data Protection Law (Law 2472/97)*, 10 April 1997, Greece.
- [EUR95] Directive 95/46, "On the protection of individuals with regard to the processing of personal data and on the free movement of such data", The European Parliament and the Council of the European Union, 24 October 1995.
- [FRO96] A. Froomkin, "Flood Control on the Information Ocean: Living with Anonymity, Digital cash and Distributed Databases", *Univ. of Pittsburgh Journal of Law and Commerce* (also available at <http://www.law.miami.edu/~froomkin/articles/oceanno.htm>), 1996.
- [GRI99] Gritzalis S., Aggelis G., Spinellis D., "Architectures for Secure Portable Executable Content", *Internet Research Journal*, Vol. 9, No. 1, 1999.
- [IMR98] IMRG Ltd., *Electronic Commerce in Europe: An action plan for the marketplace*, White Paper, July 1998.
- [ISO95] ISO/IEC 9594-8 (1994), *Open Systems Interconnection - The Directory: Authentication Framework*.
- [FRE96] Freier A., Karlton P., Kocher P., SSL ver. 3.0, Netscape Communications Corp., 1996.
- [LAW96] L. Law, S. Sabett, J. Solinas, "How to make a mint: The cryptography of anonymous electronic cash", National Security Office of Information Security Research and Technology, 18 June 1996.
- [LEE98] Berners-Lee T., Fielding R., Masinter L., Uniform Resource Identifiers (URI): Generic Syntax, August 1998 (available at <http://www.ietf.org/rfc/rfc2396.txt>).

- [W3C97] P. Hensley, M. Metral, U. Shardanand, D. Converse, M. Myers, *Proposal for an Open Profiling Standard*, W3C, 2 June 1997.
- [W3C99] L. Cranor, M. Langheinrich, M. Marchiori, J. Reagle, "The Platform for Privacy Preferences Specification", 2 November 1999 (available at <http://www.w3.org/TR>).
- [OEC98] OECD, "Implementing the OECD Privacy Guidelines in the electronic environment: Focus on the Internet", DSTI/ICCP/REG(97)6/Final, 27 May 1998.
- [OEC99] OECD, "Inventory of instruments and mechanisms contributing to the implementation and enforcement of the OECD privacy guidelines on global networks", DSTI/ICCP/REG(98) 12/Final, 19 May 1999.

This Page Intentionally Left Blank

CERTIFICATE BASED PKI AND B2B E-COMMERCE: SUITABLE MATCH OR NOT?

Kok Ming Ang

*Information Security Research Centre
School of Data Communications
Queensland University of Technology
km.ang@ieee.org*

William J. Caelli

*Information Security Research Centre
School of Data Communications
Queensland University of Technology
w.ceelli@qut.edu.au*

Abstract

This paper proposes that an urgent re-evaluation is needed to assess whether or not X.509 certificate based structures are the best technology to implement security schemes for business-to-business (B2B) electronic commerce operations. In particular it proposes that alternative structures based around simplified directory schemes and “trading partner agreements” and other concepts offer far more efficient and scalable solutions. In addition, directory structures and associated legal agreements provide a better solution to the problem of evidentiary collection and presentation in the case of disputes, particularly where these involve legal proceedings. Far more work is needed on the mirroring in information systems and data networks of the time-honoured practice of involvement of a “notary” or “witness” to an important set of transactions, such as those relevant to the B2B environment. This is markedly different to the business-to-consumer (B2C) situation involving much smaller level transactions. Overall, however, the need for trusted computing environments (such as those based around “mandatory access control” schemes) is paramount in building trust in any computer/data network scheme involved.

1. INTRODUCTION

Public key infrastructure, in support of electronic commerce, based on X.509 certificates concepts and allied technology has been extensively studied by researchers (Berkovits et al., 1994; Ellison and Schneier, 2000). However, some problems are quite visible, and many researchers have sought to use cryptographic protocols to repair flaws. Electronic media do not have the distinct features of traditionally signed paper records. Multiple digital copies are indistinguishable from each other while paper documents can be made and recorded with unique, highly unalterable characteristics. Moreover, the act of signing is itself surrounded by “ceremony” often involving one or more witnesses. Moreover, the “signer”, in approving the contents of a document through affixation of a signature, mark and/or seal, has reasonably complete knowledge of the total contents of the document to be signed and complete control over the signing process. In the case of B2B electronic commerce, problems clearly exist in these areas, particularly if commercial-off-the-shelf software systems are employed with little to no knowledge of their content or operation. These well established and legally tested processes are precisely the problems that, in many important areas such as wills and testaments, real estate titles, court records and so on, prevent electronic records from gaining complete legal recognition. Current dependence upon digital certificate structures appear to be not relevant to the solution of these important and legal requirements for trust in signing.

2. WHAT COMMERCE NEEDS

Electronic technology can satisfy business and legal requirements for the conduct of national-level and international commerce. The basic function of any electronic commerce scheme must be the ability to, at a minimum, mirror the reliability, security and trust levels developed over time through traditional commercial activities and accepted practices.

Any security scheme for B2B electronic commerce must properly address the underlying concern for business certainty. Not only does it need to cater for normal business activities, such as reliable delivery of ordered products, dependable payment mechanisms, etc, but on also the ability of the business partners to be able to resort to applicable law should a dispute occur. Traditional and accepted security mechanisms like paper trails and availability of records, business auditing, agreed contracts and signature witnessing serve to reinforce trust and certainty by provision of credible evidence of normal business practice.

Trust is an often-used but vaguely defined term. One crucial aspect is the involvement of human perception and emotion, which cannot be merely defined with mathematical or scientific rigour.

2.1. TRADITIONAL ELECTRONIC COMMERCE (EDI)

Electronic Data Interchange, or EDI, has been a strong business tool for almost three decades in a number of differing forms (Kimberley, 1991). B2B electronic commerce is just another manifestation of exactly the same business desire to more effectively perform business functions for inter-company trading while minimising the costs involved. B2B is one aspect of the broader electronic business “triangle” as shown in Figure 1.

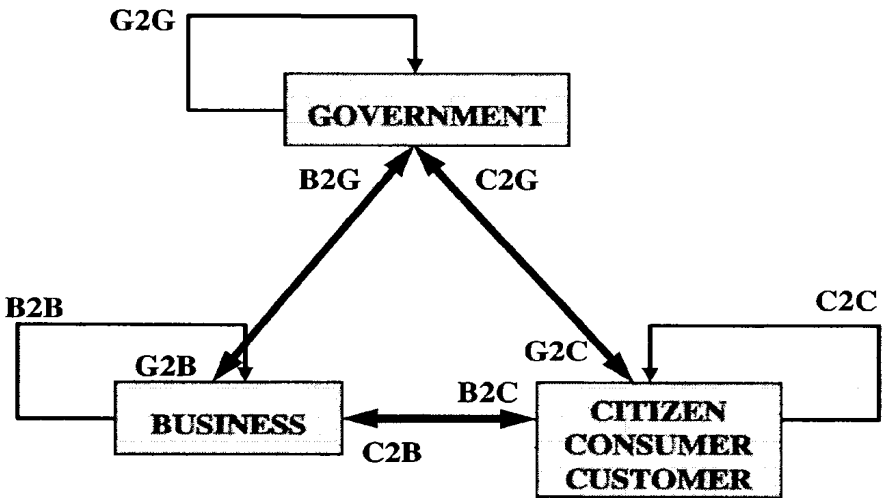


Figure 1 Electronic Business Participants

B2B electronic commerce may thus be envisaged as the latest manifestation of EDI whereby trading activities are carried out over the Internet, using its protocol suite, and because of the lowering of costs involved, now includes small to medium scale enterprises. These were often unable to avail themselves of the earlier EDI structures due to cost limitations. It should be noted, however, that other, more specialised forms of EDI have existed for a long time in specific industries such as the banking and finance industry (through EFT or Electronic Funds Transfer), etc. Kimberley (Kimberley, 1991) describes the bases of EDI as follows:

“The basic principle of EDI is that computer-generated trading documents, such as orders and invoices, are transmitted directly to a company’s trading partner’s computers across a telecommunications network. The term trading partner is used to describe any company, government department, or commercial or non-

government entity with whom an organisation regularly exchanges documents containing formatted data (i.e. not just memos or letters) as a normal consequence of carrying out business or governmental functions.”

The important principles identified by EDI include the concept of “trading partner” and the use of agreed standards for formatted data transfer.

3. BACKGROUND

EDI systems have been in place for almost two decades, particularly in Europe. Standards have emerged for their use. In particular, standards for electronic document content are vital for inter-operability, although even in the “paper world” standards assist greatly as was demonstrated (Kimberley, 1991) during the great Berlin airlift after World War II. In the USA the need for more “global” transactions standards across industries, was recognised as early as 1978 with the formation of the ANSI X. 12 committee. However, elsewhere, other routes were taken. In particular, the formation of the United Nation’s EDIFACT group in the mid-1980s was a culmination of over 10 years of work by that international organisation on facilitation of international trading procedures, that again owe their origins to the late 1940s. Standards derived in this manner found their way into the ISO scheme.

However, it is important to note that the need for security in the form of document authenticity, integrity and privacy was recognised in these EDI activities. While technological solutions were defined and the use of X.500 style directories were seen as the logical structure and place for storage of and access to required cryptographic keying materials, the use of a “Trading Partner Agreement” formed a basic concept in legal acceptance of the scheme. These agreements, which it may be argued, should still play a major role in any B2B arrangement since they give force to partner desires to trade electronically and provide a base for reconciliation and resolution should problems occur, as they invariably will.

4. HANDWRITTEN AND DIGITAL SIGNATURES

The use of the term “digital signature” in (Diffie and Hellman, 1976a) and (Diffie and Hellman, 1976b) coupled with a short explanation of the concept involved started a search for an electronic replacement for the human handwritten signature, as a verifier, through the use of the then newly re-discovered “public key cryptography” concept. However, there are significant physical and legal differences between a normal “signature” and the term “digital signature” such that the latter is not, it is contended in this paper, a straightforward replacement for use in electronic commerce.

4.1. DEFINITION OF SIGNATURES

A traditional signature is given as a mark impressed upon paper with a pen or other mechanical seal (McCullagh et al., 1998). One legal dictionary defines it as (Nygh and Butt, 1997):

Signature A person's mark on a document which indicates his or her intention to be bound by its content.

Testamentary Signature A testamentary signature may include the specific mark or initials of the testator as well as his or her name. The test for validity of the signature is whether what has been written was done by the testator as an authentication of what precedes it as his or her will.

It must be provable in a court of law that the mark is affixed with such instruments by the signing person or under his or her authorisation. In this sense, there are some notable physical and legal differences between "autographs", "signatures" and "seals". A handwritten signature is a human biometric action controlled and explicitly performed by an individual, while a seal is a physical token wielded by its owner. An autograph, interestingly, while physically resembling a traditional biometric "signature", is not a strict signature since there is no intention by the "signer" to be bound by any agreed document or the like. In addition, the legal recognition of "seals" also varies with different jurisdictions (McCullagh et al., 1998).

Assuming that "digital signature schemes" are implemented in a reliable and a reasonably trustworthy manner, complex calculations such as ' $x^y \bmod n$ ' (modular exponentiation) and message or data "hashing" are not done using mental arithmetic by a potential "signer"¹. The user has limited to no control over, and normally no knowledge of, the processes involved in the actual imprinting or "signing" act and it is therefore more logical and appropriate to rename this process as that of affixing a "digital seal". In this sense, the end user has no idea as to whether or not a digital signature created is correct at the time of affixation. He or she must have complete trust in the program used to create the digital seal and in the correct contents of computer memory at the time the "document" is "signed" or "sealed". For example, in the case of a typical home personal computer it is unreasonable to make this assumption since such systems, both hardware and software, were never designed with security requirements in mind at all.

Verifiable "digital signatures" fall under the legal definition of a more general term "electronic signature", which includes non-cryptographic markings such as digitised images and facsimiles of handwritten signatures, typed names, and

¹ This refers to the commonly used RSA digital signature exponentiation calculation.

electronic mail address headers (ABA, 1996). While electronic signatures are trivial to copy, human autographs are relatively harder, but not impossible to forge. Additional protective mechanisms have developed over time to combat handwritten forgery, such as the vital legal process of “witnessing”.

4.2. SIGNATORY EVIDENCE

In a law court, a signature on a paper document can either indicate a willingness of the signer to be bound by the document’s content or the signer’s authorship. This reliability of handwritten signatures as evidence is premised on the following assumptions (McCullagh et al., 1998):

- The signature leaves a semi-permanent mark upon the medium and cannot be easily removed without leaving any sign of alteration.
- The signature design of the person is expected to be relatively unchanging.
- The signature, or together with a printed name, can sufficiently identify the signer.

Digital media record every single bit faithfully and permit easy changes. Thus, electronic images of handwritten signatures are easily copied and unreliable and need cryptographic digital signatures coupled to them to produce a unique and unforgeable mark. Still, a digital signature on a message can be easily removed with a text editor or word processor and substituted with another different recalculated signature. This is in sharp contrast to paper, where no two signatures are exactly identical and therefore a person can be identified with his or her relatively unchanging signature pattern. Moreover, physical removal and/or substitution of a “paper signature” is still not a simple matter, even given modern imaging systems.

A digital signature is not immediately verifiable by visual inspection; its public key is required to recompute the signature from the message for verification. In turn, the public key is dependent upon its corresponding private key. The user can either claim that his private key was compromised without his knowledge and an adversary signed with his private key, or he did not authorise a computer program to sign on his behalf and the computer system did so contrary to his desire.

4.3. WITNESSING

The traditional “notarisation” process serves to counter fraud, signature forgery and repudiation (McCullagh et al., 1998). A “notary” is normally a person physically present at the act of signing to witness the physical action of the signer putting to paper an identifying mark in full knowledge of its intent, and at the same time observe the physical/psychological state of the signer and

the circumstances surrounding the act. Shortly after the signing, the notary at the scene places his/her “autograph” on the same paper as a sign of witnessing the person’s act of signing. In the event of dispute, the paper is admissible as evidence in a court of law and the notary can be called to testify on the witnessing of the signature process.

“Digital notarisatation services” provided by PKI vendors are vastly different from this traditional process. For example, Verisign’s “Validation Services” apply a digital signature and time-stamp on the document (Verisign Inc., 2000). Strictly speaking, it is conjectured that the person who is the alleged holder of the “private key” to be used for digital signing purposes, authorised a program to sign a document using that key and a third party applied another digital signature and time-stamp to the supposed signed document. Using untrusted or unreliable computers, there is a lack of reasonable proof that a document was willingly and deliberately signed by the alleged originating party, and the vendor’s authorisation marks were applied to the correct document.

Alternatively, a human notary can be physically present at the act of digital signing and apply a witnessing signature (McCullagh et al., 1998), but the lack of “trusted systems” at the home/small business and commodity computer level, again brings into question the legal validity and certainty of such actions. It appears obvious that any usage could be reasonably open to challenge in a court of law in the case of dispute. Neither side to a court action could present irrefutable evidence that the computer systems used was reasonably protected against tampering, insertion of “Trojan Horse” or “viral” programs, untrustworthy or unreliable software sub-systems, protection of the signing process and the associated cryptographic keys, etc.

4.4. CEREMONY

The process of signing on physical medium carries a cautionary purpose. The signer’s attention is brought to the gravity of a document's contents and its likely legal consequences (Jueneman and Robertson, 1998, pp. 430-431). By signing the document, the person is presumed to have understood its contents and is therefore willing to accept its terms. The psychological burden is more pronounced in the presence of witnesses and is often adequate to deter hasty signing. Even with seals, such as usage in the case of “deeds”, etc., this ceremonial importance in contract approval is notable. In the earlier case of EDI, mentioned above, this ceremonial function was largely taken up by the legally binding “Trading Agreement” that covered all activities between the parties to an EDI scheme.

It could be argued that frequent users of computers and similar devices have developed “Pavlovian” behavioural characteristics such as repetitive clicking on graphical window menu items and buttons (Sneddon, 1998), particularly where

the consequences of the action are incompletely understood and the underlying technology base is “foreign” to them. Automated batch programs and command scripts are also used to take the drudgery out of predictable computer input and responses. Unlike writing a autograph, there is a lack of ceremony in using computer interfaces for digital signing. This comparative social difference may be contested by users who are not aware that an act of signing has been inadvertently committed on their behalf.

4.5. BURDEN OF PROOF

Under the common law, a person has the right to deny a signature that is attributed to him or her (McCullagh and Caelli, 2000). The fact-finder or “relying party” will have to supply sufficient evidence to prove the signature’s authenticity or the valid circumstances surrounding the signature’s formation.

However, legislation on “electronic: transactions” appears to have taken a different step. The United Nations Commission on International Trade Law (UNCITRAL) Article 13 (McCullagh and Caelli, 2000) and the Utah Digital Signature Act (Biddle, 1996) attempt to shift the burden of proof onto the signer. This departure from traditional legal norms does not account for a number of problems.

In traditional signing, the signer has total control over his or her signing action and does not need to worry about any other mechanism that may falsely insert, steal or record the signature². On the other hand, in the electronic case, computer viruses, smartcard physical theft or computer hacking can compromise the signer’s private key, a vital component of the signing process. Under the Utah Act, the signer must prove that the signature was not affixed by himself or herself and sufficient duty of care was exercised, although the Utah Act remains silent on what constitutes reasonable care (Biddle, 1996).

Recent legislation on electronic commerce such as the E-Sign Act (E-Sign Act, 2000) accord legal recognition upon electronic records and signatures. However, the inconsistent recognition of non-repudiation issues between paper and electronic records may hamper the paper-to-electronic commerce transition, or even electronic commerce across different social-legal borders.

5 . MISCONCEPTIONS

The words “digital” and "electronic" are frequently used interchangeably. As a result, many laymen are confused over the differences between digital and electronic signatures, and policy makers often mistake the former definition for the latter. Nevertheless, digital signatures are still different from the traditional

²Although carbon paper can also duplicate signatures, it is relatively easy to detect its use. (McCullagh and Caelli, 2000)

version. Even with biometrics incorporated into digital signature processes (Jueneman and Robertson, 1998) and associated timestamping techniques, these do not have the affirmative features of witnessed, written signatures. A digital signature (or seal) cannot be equivalent to a written signature (Harbison, 1998, p. 114), even if laws are passed to try to make it so.

5.1. THE PURPOSE OF CERTIFICATION

An original proposed application of public key cryptography was to create a secure directory service to assist communication privacy between users and prevent impersonation attacks (Diffie and Hellman, 1976b; Kohnfelder, 1978). This required a user to contact an opposite party, pause communication while the pertinent public key of that opposite party is retrieved from a directory and then verified, and then proceed on with secure communications, which was inconvenient (Kohnfelder, 1978, p. 39). In addition, the administrative burden of maintaining a large, secure, database of people's public keys is difficult. Hence, the concept of "digital certificates" was proposed and designed to reduce the need for frequent public key retrievals and associated directory updates.

However, Kohnfelder also admitted that certificates would not provide any extra benefit when the directory is compromised or users frequently lose their keys (Kohnfelder, 1978, p. 42). In these situations, the costs of certificate revocation outweigh the benefits of certificate use. This is contrary to the belief that certificates provide ". . . a scalable and secure method (from an integrity perspective)" to distribute public keys (Adams and Lloyd, 1999, p. 74). Certificate may even add a greater administrative burden for directories maintenance and users with little to no advantage at all.

5.2. DIRECTORY SERVICES

Directories are not suitable for holding, or ever intended to hold, the private cryptographic keys used for digital signing purposes. They were originally meant to store communications secrecy keys, such as "session keys", and not signing keys (Diffie and Hellman, 1976b). When a directory user dials a wrong telephone number or sends an encrypted message to the wrong person, it is merely an inconvenience. Nobody sues a telephone directory publisher for wrong information (Landrock, 1999, p. 411), because there is no associated legal burden.

5.3. UNTRUSTED COMPUTING

One major problem common to all security woes is the lack of trustworthy and reliable computing systems (Thompson, 1984; Anderson, 1994b; Anderson, 1994a). The functions of a secure information processing system requires authentication, authorisation and detection and compensation of non-orderly

behaviour including software and hardware reliability and human behaviour (Dierstein, 1990).

A computer system, or its cryptographic software, may be compromised by viruses or Trojan horses with no tell-tale signs. Private keys or pass-phrases to these keys can be stolen and be used to falsify document authenticity. Flaws in computer hardware, operating systems and cryptographic software become a burden on the end user. Without any legal liabilities at present, manufacturers see little need to take remedial steps and to offer high-trust commodity computer systems. Computer hardware manufacturers and software houses are not mentioned in legislation as maintaining any liability, and legal disclaimers place the risk of computing systems onto users.

It is almost impossible to dictate user key management practices (Anderson, 1994b; Davis, 1996). Responsibility for the protection of a user's private key, essentially their "digital identity", lies solely with the user. This is clearly unreasonable where available computing systems and the public key certification systems does not adequately address the needs for higher level access control, such as "mandatory access control schemes".

Fraud does occur with paper based B2B commerce systems, but control mechanisms have been developed over time by society, governments and the legal system to deal with it. Although it is trivial to forge letters, paper audit trails and extensive record keeping help reduce forgery. An executive within a company can exceed his or her powers and perform an unauthorised and potentially illegal transaction on behalf of the company. In a court of law, the organisation of the executive is responsible for the said transaction, and, in turn, could press criminal charges against the erroneous executive.

There is no trustworthy path from the user to the end of the communication path. Paper-based systems employ paper trails and audit practices, while digital signing processes lack such multi-faceted structures and do not provide easily recorded and dependable forensic evidence. Also, there is no control over the digital signing process, and software cryptographic processing cannot be observed, much less understood, by end users. Thus, commercial computing systems are not suitable to be held as good evidence in a civil dispute whose resolution depends upon the weighing of the "balance of probabilities" (McCullagh and Caelli, 2000).

5.4. THE ABSENCE OF "ROLES"

There have been many discussions amongst researchers on the problem of delegating responsibility and trust (Crispo, 1998; Harbison, 1998). What is not addressed properly is the recognition of roles, departmentalisation of organisations and division of job functions which has existed since Biblical times (The Holy Bible, 1984; Stoner et al., 1997). A job position or role is maintained by

an organisation. A member or employee of the organisation is appointed to fill the role, and the appointed person may change over time. A signature made by the person at a point in time is performed under the authority of the position within the organisation.

Commercial paper documents and contracts normally display the originating organisation's letterhead, the job position of the signer and his or her name. The handwritten signature on the letter serves to authenticate the name (and not the other way around), and the position implies the authorisation accorded by the organisation. The receiver can check if the signer is the correct and authorised organisation member, and check with the state or national business registry to verify the legitimacy of the organisation. Therefore, it is strange to base business to business trust on the verification of a public key certificate of a signer provided by a third party (CA) that does not know anything about an employee's position or organisation's purpose. In normal business these should be accomplished by recourse to the organisation or to a business registry, respectively.

In the normal B2B commerce case, verification of the authority and validity of a document is done with the organisation or department in question on a need-to-know basis, clearly a more efficient and flexible as well as time-honoured practice. A hierarchical, X.509 certificate based PKI attempts to act as a large, distributed Access Control List (ACL) for individual end-user entities. Certificate revocation is meant to be broadcast throughout the PKI, which is difficult to carry out (Davis, 1996). In contrast, in a normal business case, when a person leaves an organisation, centralised information servers revoke the person's authorisation and privileges. This does not require a broadcast to the entire organisation or to those outside it, such as trading partners.

A PKI structured on the CA and X.509 certificate concepts does not and cannot provide an universal signing function for various business and social purposes. Digital signatures that are not tied to a particular context are meaningless (Feigenbaum, 1998). It is highly possible that a certificate verifier (user) only sees the correct verification of an electronic purchase order signed by an Adam Smith of ACME Corporation without realising that Smith is actually a rogue ACME system administrator. This scenario is compounded by the fact that CAs are not concerned about a "certified" signer's authority to sign any particular document.

In the B2B context, an organisation handles and bears the authority of signing, while this authority is then delegated onto designated individuals. Authorisation mechanisms such as Simple Distributed Security Infrastructure (SDSI) (Rivest, 1998), credential certificates (Ellison, 1999) recognise the need for delegation of authority, but they do not recognise the purpose of roles, which are essentially privileges and restrictions of a user, and are prevalent in access control literature.

A person is assigned into a particular role in an organisation, and is authorised by his organisation to sign or delegate executive authority to certain colleagues.

Consequently, public key signing functions need to be integrated into access control systems, as much as employee responsibilities and authority are fitted into roles in a structured corporate hierarchy.

6. NEW PROPOSALS

Corporations are responsible for delegating the responsibilities and executive powers of its personnel. In a similar way, national business registration authorities track the existence of commercial enterprises. Professional organisations, such as medical, accounting and legal councils regulate their respective practitioners. Certification by respective authorities would probably be more trusted and recognised than a CA that issues generic certificates.

6.1. “RELIABLE” CERTIFYING AGENCIES

Evidently, at the moment such authorities are not yet utilised in the digital domain, e.g. Internet domain name registration, etc. It is suggested that they are in a far better position than commercial vendors to provide the social authority desired in commerce and industry.

Across national borders, cross verification between such authorities could be in a “web of trust” structure, where the number of links maintained by each authority in its domain of interest is approximately limited to the number of participating United Nation countries.

The argument is that an existing social authority or entity should see itself as a digital “certification” authority. (This is not the same as a current CA providing X.509 digital certificate services).

Governments would, at a minimum, play a regulatory role to provide trustworthy, verified mechanisms. A governmental department could be created to manage public cryptographic keys, in the same manner that a government does today through issuance of a passport, registration of companies, etc. A national government has a more enduring permanence than corporations, which may be subject to more common dissolution. Such a government agency would aim to provide a critical function in society, as opposed to a CA where profit is its primary purpose.

6.2. DIGITAL SIGNING WITH ROLES

There is a natural gap between the world view of an organisation and its actual internal structure and management. It is obvious that organisations normally run their own business and assign roles to their members to accomplish business or like objectives. Ideally, the use of Role-Based Access Control (RBAC) mechanisms in information systems may be used to control digital signing activities. This indeed may correspond to the “name and position” title structure of a authorising signature on a commercial, paper document.

Employees who are vested in the roles are given a set of keys stored on tamper-resistant hardware such as smart cards that may now be used to act for the company or entity. In the event that an employee is unable to be in the office, for example, due to illness, accident or death, another employee can take over that job function without significantly undermining the operation of the role. Delegation thus becomes a normal part of the B2B e-commerce structure, mirroring usual business practice.

Figure 2 shows the order in which an electronic message is signed by Bob with his own personal key, followed by the role CEO and the company ACME Corp. (where the role and company signing are performed by separate, trusted computers). In the event that Bob is away for a meeting, a delegated manager Alice can sign on Bob's behalf, and she would be potentially responsible for any discrepancies. The activation of role and/or company keys by personnel or machine are left to individual corporate policies.

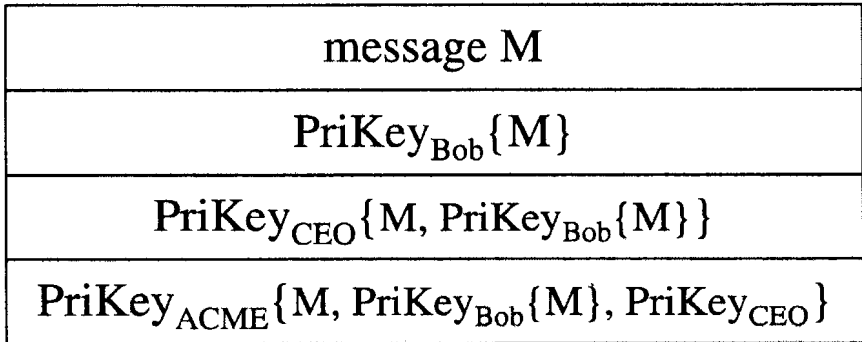


Figure 2 Proposed Digital Signature Hierarchy

Company keys can be certified by a business registry, role keys by the organisation, and individual keys by the organisation's human resources department. Such a scheme mirrors paper practices, and its simplicity makes it easier for laymen to adopt such an electronic parallel. Widespread adoption of this method may encourage the growth of B2B PKI-based electronic commerce.

7. CONCLUSION

This paper presents a simple approach to overcoming excessive dependence on X.509 digital certificate structures for B2B electronic commerce activity. Commerce demands reliable and safe methods, particularly where conflict resolution is needed via mediated negotiation or legal recourse, which complex certificate-based PKI structures have failed to provide.

The solution lies in a combination of technical and organisational structures, as follows:

- 1 Recognition that complex X.509 digital certificate hierarchies and/or networks do not totally meet the needs for efficient and reliable B2B e-commerce demands,
- 2 Alternative structures based around simplified directory schemes and “trading partner agreements” and other concepts offer far more efficient and scalable solutions,
- 3 Directory structures and associated legal agreements provide a better solution to the problem of evidentiary collection and presentation in the case of disputes, particularly where these involve legal proceedings,
- 4 Far more work is needed on the mirroring in information systems and data networks of the time-honoured practice of involvement of a “notary” or “witness” to an important set of transactions, such as those relevant to the B2B environment as distinct to the business-to-consumer (B2C) situation involving much smaller level transactions.

The crux of the infrastructure issue in B2B electronic commerce is not simply concerned with advanced cryptographic or security techniques. It is about providing electronic services and mechanisms that are at least equivalent to, and hopefully superior to, current social and legal rules deeply embedded in human society.

References

- ABA (1996). *Digital Signature Guidelines*. American Bar Association.
- Adams, C. and Lloyd, S. (1999). *Understanding Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations*. Macmillan Technical Publishing, Indianapolis, Indiana.
- Anderson, R. J. (1994a). Liability and Computer Security - Nine Principles. In *Proceedings of the Third ESORICS*, volume 875 of *Lecture Notes in Computer Science*, pages 231–245. Springer-Verlag.
- Anderson, R. J. (1994b). Why Cryptosystems Fail. *Communications of the ACM*, 37:32–40.
- Berkovits, S., Chokhani, S., Furlong, J. A., Geiter, J. A., and Guild, J. C. (1994). Public key infrastructure study: Final report. Technical report, Mitre Corporation.
- Biddle, C. B. (1996). Misplaced Priorities: The Utah Digital Signature Act and Liability Allocation in a Public Key Infrastructure. Unpublished but submitted to *San Diego Law Review* Vol. 33.
- Crispo, B. (1998). Delegation of Responsibilities (Transcript of Discussion). In *6th International Workshop on Security Protocols*, volume 1318 of *Lecture Notes in Computer Science*, pages 124–130. Springer-Verlag.

- Davis, D. (1996). Compliance Defects in Public Key Cryptography. In *Proceedings 6th USENIX Security Symposium*, pages 171–178, San Jose, California. Usenix Association.
- Dierstein, R. (1990). The Concept of Secure Information Processing Systems and Their Basic Functions. In *Proceedings of 6th IFIP/Sec'90*, pages 133–149, Helsinki, Finland. North Holland.
- Diffie, W. and Hellman, M. (1976a). Multiuser Cryptographic Techniques. In *Proceedings of AFIPS National Computer Conference*, pages 109–112. AFIPS.
- Diffie, W. and Hellman, M. (1976b). New Directions in Cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654.
- E-Sign Act (2000). Electronic Signatures in Global and National Commerce Act. Second session of the 106th Congress of the United States of America on 24 January 2000.
- Ellison, C. (1999). SPKI Requirements. RFC 2692.
- Ellison, C. and Schneier, B. (2000). Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. *Computer Security Journal*, 16(1):1–7.
- Feigenbaum, J. (1998). Towards an Infrastructure for Authorization. In *3rd Usenix Workshop on Electronic Commerce*, pages 15–19, Boston, Massachusetts. USENIX Association.
- Harbison, W. S. (1998). Delegating Trust (Transcript of Discussion). In *6th International Workshop on Security Protocols*, volume 1318 of *Lecture Notes in Computer Science*, pages 108–117. Springer-Verlag.
- Jueneman, R. R. and Robertson, Jr., R. J. (1998). Biometrics and Digital Signatures in Electronic Commerce. *Jurimetrics Journal of Law, Science and Technology*, 38:427–457.
- Kimberley, P. (1991). *Electronic Data Interchange : A Review of the Current Status of Electronic Data Interchange Throughout the World and an Introduction to the Services*. McGraw-Hill.
- Kohnfelder, L. M. (1978). Towards a Practical Public-key Cryptosystem. Bachelor's thesis, Department of Electrical Engineering, MIT.
- Landrock, P. (1999). Challenging the conventional view of PKI - will it really work? In *Proceedings of 16th World Conference on Computer Security, Audit & Control*, pages 406–424, Westminster, London. Elsevier.
- McCullagh, A. and Caelli, W. J. (2000). Non-Repudiation in the Digital Environment. *First Monday*, 5(8).
- McCullagh, A., Little, P., and Caelli, W. (1998). Electronic Signatures: Understand the Past to develop the Future. *University of NSW Law Journal*, 21(2).

- Nygh, P. E. and Butt, P., editors (1997). *Butterworths Australian Legal Dictionary*. Butterworths, Sydney.
- Rivest, R. L. (1998). Can We Eliminate Certificate Revocation Lists? In *Proceedings of Financial Cryptography '98*, pages 178–183, Berlin. Springer-Verlag.
- Sneddon, M. (1998). Legislating to Facilitate Electronic Signatures and Records: Exceptions, Standards and the Impact on the Statute Book. *University of NSW Law Journal*, 21(2).
- Stoner, J. A. F., Yetton, P. W., Criag, J. F., and Johnston, K. D. (1997). *Management*. Prentice-Hall, Sydney, Australia, second edition.
- The Holy Bible (1984). *The Holy Bible*. International Bible Society, East Brunswick, New Jersey, New International Version edition. Exodus Ch. 18 v. 13–23.
- Thompson, K. (1984). Reflections on Trusting Trust. *Communications of the ACM*, 27(8):761–763.
- Verisign Inc. (2000). Verisign OnSite White Paper. Available at http://www.verisign.com/onsite/white_paper.html.

Acknowledgments

The authors wish to thank their colleagues in the Information Research Security Centre, Queensland University of Technology for the lively discussions on this topic.

Kok Ming Ang graduated with a Honours degree in the Information Security Research Centre (ISRC) at the Queensland University of Technology, Brisbane, Queensland, Australia.

William J. Caelli is the Head of the School of Data Communications and a Member of the Information Security Research Centre (ISRC) at the Queensland University of Technology, Brisbane, Queensland, Australia. Prof Caelli is the research supervisor for Mr Ang.

INTERNET ANONYMITY: PROBLEMS AND SOLUTIONS

Claudia Eckert and Alexander Pircher
*Faculty of Mathematics and Computer Science,
University of Bremen, Germany*
eckert@informatik.uni-bremen.de

Abstract Internet services like the World Wide Web or email programs are already widely in use for private and business work. Unfortunately, with every access a lot of user-specific information is leaked. Hence, using popular Internet-services results in threats against user privacy, as this data can be eavesdropped by attackers or collected by service providers in order to create user profiles. To defeat such threats anonymizer services have been introduced especially for anonymous email and net news. But the available anonymizing services lack a lot of deficiencies and do not provide the required degree of anonymity to Internet users. This is mainly because these services have been implemented in a rather ad hoc manner lacking a systematic analysis.

The Anonymous-project aimed at revealing and overcoming the deficiencies of existing approaches by following a systematic methodology. Our paper summarizes the main results of the Anonymous-project. It explains the problems and limitations of current anonymizing services and presents our new services.

Keywords: Anonymity, Privacy, Internet Protocols

1. INTRODUCTION

Internet services like the World Wide Web or email programs are already widely in use for private and business work. Unfortunately, with every network access a lot of user-specific information is transmitted over public networks. Examples of such information are the user's IP address, the URL of the previously loaded Web page or date and time of the performed access. Hence, using popular Internet services results in threats against user privacy, as this data can be eavesdropped by attackers or collected by service providers in order to create user profiles. The problem is that the user normally does not know to which extent sensitive data concerning his privacy is collected and stored.

For instance, let us have a look at ordinary emails. Normally, a sender will be willing to give away his own email address, but he certainly does not want to reveal other information like the URL of his previously visited Web page.

Until now, several programs for anonymizing Internet services have been developed. They use different techniques with respect to data avoidance and/or data concealment. Usually, data avoidance is achieved by suppressing the relevant data. Data encryption provides appropriate means to conceal transferred data preventing unauthorized third parties from accessing plaintext information. But data encryption is not applicable for those data items that must be accessible by the authorized communication partner to be able to correctly execute the used transfer protocol (e.g. HTTP, SMTP). Data hiding in such scenarios can be achieved by replacing them with some uniform patterns. The problem is, that the existing anonymizing services just hide or anonymize parts of the sensitive data. The result is that Internet users are not as anonym as they could be. Furthermore, in some services the replacement of original data by uniform patterns results in faults during protocol execution.

All these deficiencies discovered in existing services can be prevented if service development is performed based on a systematic analysis of the Internet-services. Analyzing the protocol specifications of Internet standard protocols enables to classify the transferred data into the class of sensitive data which must be anonymized and all the rest which is not critical. In addition, a serious analysis can reveal to which extent the sensitive data is in fact required to perform the protocol successfully. This has been done within our Anonymous-project [5]. Our aim was to repair the discovered deficiencies by developing anonymizing services which are stronger and more flexible than the existing ones.

The rest of our paper is organized as follows. Section 1.1 presents the main results of our investigation of common Internet services. To be able to compare the strength of anonymizing services we have developed a simple metric which we will explain in section 1.2. Based on our analysis and the metric we have evaluated existing anonymizing services. The results are summarized in section 2. Based on our experiences we have developed our own suite of anonymizing services which will finally be presented in 3. In 4 we summarize the main contributions of our paper.

1.1. ANALYSIS OF RELEVANT INTERNET SERVICES

Popular Internet services like HTTP, FTP, SMTP, or NNTP are client-server applications running on top of the Internet protocol (IP). These services deliver data packets to the IP layer which appends an IP header to each packet. The IP header contains among other data the IP address of the packet's sender and receiver. Hence, an anonymizing service could simply try to conceal the

sender's IP address. But this simple technique is not applicable if the protocol requires a bidirectional communication between sender and receiver or if the protocol must send messages back to the sender, for instance, in cases of faults. In addition, we should notice that IP addresses are often already appended to data packets on higher protocol layers by application protocols themselves. This is why address data should be filtered on a per application protocol basis in addition to the IP level filtering. As all the investigated services have in common that anonymity with respect to third parties can be achieved by incorporating encryption techniques, we omit this technique in our subsequent discussion.

1.1.1 HTTP (Hypertext Transfer Protocol).

HTTP [3] is a request/response protocol which establishes a bidirectional connection between client and server. A request message of a client consists of several headers and the message payload. Classifying the header data into privacy critical and uncritical data we observe that the *referer*-entry obviously belongs to the first class. This entry contains the URL of the page from where the server was called. Hence, the server is able to gain context information about his clients by simply inspecting the *referer*-field. Consider for example a client who sends a request in the context of a search engine. In this case, the *referer*-entry contains the whole hit list of the search previously performed by the client. From the server point of view this list might contain interesting information about competitors in the digital market. In addition, the searched key words transferred within the *referer*-entry reveals a lot of information about the client's intends and requirements.

The various *accept*-fields used in HTTP messages belong to the class of sensitive data as well. These fields normally contain the preferred character sets, the language and coding schemas etc. of the client. By carefully analyzing the *accept*-fields an adversary is able to derive a lot of critical information concerning the HTTP client. IP address information can be found in the fields *client-IP*, *X-Forwarded* and *cache control* which therefore belong to the class of sensitive information, too. Another sensitive field is the *user-agent*-field which identifies the client's user-agent and the optional *from*-entry specifies the email address of the user who is associated with the user-agent. Authentication information is transferred within the *authorization* and *proxy-authorization*- fields. They might contain the user name and user password which is usually just base64 coded. Besides all these headers that are fully specified by the protocol specification, a HTTP message might possess headers which can be defined by users in arbitrary manner, possibly containing lots of user-specific sensitive information.

In contrast to the header fields discussed above which are created by the client system the *cooky*-entry contains data that has originally been created by the server and is stored on the client side. The *cooky* is transferred automatically by the client-browser whenever the client re-connects to the server. The transferred

information enables the server to re-identify the client though the underlying HTTP protocol is stateless. The problem is, that usually the client does not know what information is sent to the server encoded in the *cookie*-entry. As a client often reveals privacy related information about himself by filling in Web-forms delivered by the server, the server can extract user-specific information and encode them in cookies. Each time the client re-establishes a connection to the server, the server just decodes the cookie to reveal the identity of the client.

Anonymizing HTTP

We have analyzed 7648 Web requests. First, we observed that in 99% of all of these requests the *accept*-fields contained the above mentioned sensitive data but which was not used by the receivers at all. Hence, within an anonymizing service this data can be either completely avoided or substituted by other patterns without disturbing the server's functionality. In contrast, the information within the *user-agent*-field which was present in over 98% of the requests, actually was used by the servers to tailor the presentation of the required pages to the specific capabilities of the client's browsers. Hence, services which aim at anonymizing these data should be configurable in a flexible manner. This will allow to use the tailored services even in the context of anonymizing activities.

Information contained in the *referer*-field have been transferred in more than 95% of all analyzed requests. This is remarkable, because that information is not necessary to execute HTTP correctly. As we have pointed out previously, an adversary might infer a lot of sensitive information about the sender looking at the *referer*-field. Hence, we recommend to omit all data within *referer*-fields.

Our analysis revealed that the data in the *from*-fields are neither used by the protocol itself nor by service providers. As it might contain a lot of interesting information for an unauthorized third party, we strongly recommend to omit these fields as well, Cookies have been observed in at least 30% of all analyzed requests, though the requested services are usable correctly without them as well. Hence, leaving out cookie data by anonymizing services will not cause an unacceptable denial-of-service.

To avoid the unprotected transfer of IP address information within header fields of HTTP, we recommend to anonymize the fields *client-IP*, *X-forwarded-for* and *cache control*. This is feasible without disturbing the overall protocol functionality, because these fields are not required in order to execute the protocol correctly. Since the header field *proxy-authorization* is solely required to authenticate the browser with respect to the proxy server, we recommend that the proxy should anonymize this information before forwarding the modified message. Finally, we require that a HTTP-anonymizer should be configurable in such a way that all unknown headers will be anonymized by default. But the anonymizer should be flexible enough to allow selectively an unconcealed transfer of such header data.

1.1.2 FTP (File Transfer Protocol).

FTP [7] provides services to efficiently transfer data. It works session-based and differentiates between control and payload data. A separate control channel is established to transfer control data and this connection is held open during the whole session. In contrast, FTP establishes a new connection for every send or receive transaction for transferring payload data. Notice, that each time FTP establishes a connection either for control or data transfer, the IP address of the client is transmitted. In addition, some commands require parameters which might contain sensitive information. These commands are the *user*-command which identifies the user via an ASCII-string, the *pass*-command which contains the user-password as a parameter, and the *acct*-command requiring a parameter that identifies the user's account name. Using the anonymous FTP service, current Internet browsers normally provide a password that identifies the used browser type (e.g. mozilla@ in the Netscape Communicator). Besides, browsers usually possess the option to transfer the email address of the user which is registered in the browser configuration file.

Anonymizing FTP

Since non-anonymous FTP requires the user name, password as well as user account name to be able to authenticate the FTP-client anonymizing this data at the side of the communication partner (i.e. the FTP-server) is not possible. The same holds for the client's IP address which is required to establish data connections. As mentioned before, encrypting all these data is appropriate to thwart attacks from unauthorized third parties. With respect to anonymous FTP we recommend to anonymize the transferred email address. This can be accomplished for instance by substituting it with a fictive one concealing the true identity of the FTP user.

1.1.3 SMTP (Simple Mail Transfer Protocol).

As we are aware, that there are a lot of good arguments that doubt the appropriateness of anonymous emails we subsequently just focus on one special aspect which we think is an important one. That is, in our opinion an email sender should reveal his identity to his receiver(s), but besides this, he will not be willing to expose any other information concerning his privacy. Examples for such information that should be suppressed are data about the used execution environment like the operating system and hardware. Even more important is the suppression of data about the sending context like links to other messages the email refers to.

The transfer of emails is the main task of SMTP [6]. To this end, SMTP establishes a bidirectional connection between client and server. This connection is afterwards used to transfer several mails. Analogous to FTP SMTP requires the client's IP address for establishing this connection and further sensitive data can be transferred by calling specific SMTP commands. The *mail*-command spec-

ifies a parameter that contains the return path to the email sender. This path is used in cases of faults to resend the mail to its sender. The *data*-command is used to transfer the mail data. The data itself consists of a header part and a payload part. One in our sense important header item is the *return-path*. It specifies the sender's address as well as the return path to this address. This path is constructed as follows. Every sender who takes part in transferring the mail to the final destination, appends its own address as well as date and time information to the path. Hence, the path shows in detail the whole route over which the mail was routed to its receiver.

The sender's identity can be derived from other header items as well. Critical in our sense are the *from*, *sender* and the *reply-to* entries. The *from* field contains the identity of the sending agent (e.g. a machine or a person). The *sender* entry contains the identity of the sender in cases where the author of the message is not its sender. The *reply-to* field specifies the mail boxes to which answers can be send. Furthermore, the *in-reply-to* and *references* header fields contain problematic data as well. They identify the mail to which the current mail replies as well as all other messages that are referenced in the mail.

Besides the fields that are specified in the SMTP standard a mail might contain arbitrary user-defined as well as so called extended header (starting with X-) fields.

Anonymizing SMTP

Our analysis revealed that most of the header fields contained in mails are not required by SMTP. Hence, we recommend to anonymize all header items that carry information beyond direct sender identification. That is, at least the *in-reply-to*, *return-path* and *references* field should be anonymized to conceal critical data as far as possible not only with respect to third parties but with respect to authorized email receivers as well. As mentioned before, some of these headers are in fact useful in case of trouble shooting. Therefore, the anonymizing service should be flexible enough to allow for individual configurations.

Since extended headers characterize the sender quite good and, in addition, might contain arbitrary data we strongly recommend to suppress all these non-standardized but widely used headers. If complete suppression is not feasible the data should be replaced by random patterns. This will not disturb the execution of SMTP because the protocol does not require the extended headers. Obviously, unknown headers (user-defined) should be completely suppressed by anonymizer.

1.1.4 NNTP (Network News Transfer Protocol).

As before, we do not argue in favor for anonymous postings, but focus on the intension to restrict the transfer of sensitive data to a minimum. That is, we are especially interested in such data that is automatically appended to a posting (or

mail, see previous section) without giving the user any opportunity to control or regulate this.

NNTP [4] offers services to read, post and distribute news articles. To this end, the client establishes a bidirectional connection to the news server. This connection is then usable for several actions like reading and posting articles. Like the SMTP protocol NNTP specifies header fields for news articles. These headers are comparable to those used in SMTP and need not be discussed again. In addition, a news header can contain a field called *organization*. This field identifies the organization to which the sender belongs. It therefore carries interesting data that might characterize the sender quite good. Analogous to HTTP and SMTP NNTP allows to use non-standardized headers which might contain arbitrary user-specific data.

Anonymizing NNTP

Because of the similarities between SMTP and NNTP we recommend similar anonymizing actions. That is, an anonymizer should conceal or suppress all automatically generated header data. Again it would be helpful, if the anonymizer is flexible enough to selectively de-anonymize data items. None of the non-standardized headers are required for the correct functionality of the protocol. Hence, all these potential dangerous headers should be anonymized by default. But since some of these data might be used by some servers it should be possible to selectively de-anonymize the required set of data.

1.2. LEVELS OF ANONYMITY

The subsection presents a simple metric to compare the strength of anonymizing services. First, we want to capture the notion of anonymity more precisely. *Anonymizing* in our sense means the modification of privacy-related data with the aim that single data carrying privacy-critical data can no longer be associated with a specific person except a huge amount of money, time and man power will be spent.

A weaker form is given by *pseudo-anonymity*. Here, privacy-related data is modified according to a specific assignment rule (usually by using pseudonyms) with the effect that single data carrying privacy-critical data can no longer be associated with a specific person without the knowledge of the assignment rule.

Now, we will define several levels of anonymity to be able to distinguish between anonymizing services of different strength. The different strengths of the levels result from the different scopes of the anonymizing measures. We distinguish between the following three scopes: anonymity (1) against the communication partner, (2) against third parties, and against (3) the anonymizer itself. The strongest form of anonymity is provided, if the anonymizing service covers all three scopes.

Level 1 Pseudo-anonymity with respect to the communication partner:

The communication partner is not able to associate single data items with a specific person. But all measures to achieve pseudo-anonymity are solely performed by the communication partner himself. That is, this kind of anonymizing can neither be influenced by the user nor is he able to control the success and the correctness of the service. On the other side, the user is completely relieved from coping with anonymizing actions. Services on this level do not provide anonymity with respect to third parties.

Level 2 Pseudo-anonymity with respect to the anonymizing service:

In contrast to level 1, privacy-critical data is only transferred to the anonymizing service and not to the communication partner, which provides a higher level of anonymity. The communication partner solely knows a pseudonym of the user without being able to associate it with a real person. But using such an anonymizing service is not fully transparent for users as they must explicitly call it. And, as before, services on this level do not provide anonymity with respect to third parties.

Level 3 Anonymity with respect to the communication partner:

Analogous to level 1 all anonymizing measures are performed by the communication partner without being controlled by the user. No anonymity with respect to third parties is provided.

Level 4 Anonymity with respect to the anonymizing service:

Analogous to level 2 privacy-critical data are anonymized by a third party before being delivered to the communication partner. Again we do not have a protection against third parties.

Level 5 Anonymity with respect to third parties:

The user is protected against the communication partner as well as against intermediate third parties.

Level 6 Anonymity with respect to the anonymizer:

Up to level 5 the anonymizer always possesses knowledge about its users as well as their communication partners. On this level we require that the anonymizer is not able to infer a connection between users and communication endpoints. In addition, the level requires anonymity against intermediate third parties.

2. ANONYMIZER – STATE-OF-THE-ART

Most of the activities in the area of Internet anonymity concentrate on email and news anonymity whereas anonymizing services for the WWW and FTP area are hardly available. Since we are interested in anonymity of client-related data, we do not investigate projects that concentrate on server anonymity like the JANUS-project [2].

2.1. WWW AND FTP

The most simple anonymizing services just anonymize the log file of WWW-server (e.g. <http://www.media.mit.edu/~daniels/software/scramble.html>). We call such anonymizer **log file anonymizer**. The log file records all the accesses to the server. If the log file anonymizer is integrated into the Web-server then the data can be recorded in the log file in an already anonymized form. Depending on the fact whether anonymity or pseudo-anonymity is offered, such anonymizers only provide level 3 or level 1 services with all the problems mentioned above. Even worse, the measures incorporated in existing approaches are faulty in such a sense that not all critical data is really anonymized. To be more concrete, these approaches just conceal the host name, IP addresss and user name. But our analysis (see the previous section) revealed that a lot of other data fields containing privacy-critical data still exist, like for example the *referer* or *user-agent* fields. This data is stored unconcealed in the log file of the server.

Proxy-Server

Other implementations of WWW or FTP anonymizer integrate their service into a proxy server. A popular example for this kind of anonymizing technologies is the *Junkbuster* (<http://www.junkbuster.com>). Using the proxy approach allows to modify or conceal critical data before the messages are delivered to their destinations. For instance, the Junkbuster suppresses the forwarding of cookies, and of *from*- as well as *referer*-fields and it substitutes user-agent data uniformly by *Mozilla/3.01 Gold*. But, with Junkbuster all other fields containing critical data like the *accept*-fields are forwarded to the final destination without modification. Especially, all unknown header fields are forwarded without being anonymized. Hence, the anonymizing service offered by Junkbuster is incomplete. Since the proxy-integrated anonymizer do not encrypt the data transfer between browser and proxy-server, such anonymizer could only be classified to level 4 or just 2 in ease of pseudo-anonymity.

Web-Anonymizer

Web-Anonymizer work quite similar to proxy-integrated anonymizer. One popular representative is the *Anonymizer* (<http://www.Anonymizer.com>). But in contrast to the proxy approach, the required services of Web-anonymizer are integrated into the Web-server. As a result, this service can be used behind a firewall and the data between browser and Web-server can be transfered in encrypted form. Unfortunately, these advantages are accompanied by an additional management overhead compared with proxy-approach. Additional anonymizing activities are required if a Web-page or a file that has been requested by a client does itself contain references. If the client clicks on these references directly no anonymizing services would be applied to these Web accesses. Hence, such references must be modified to ensure that each call (clicking on the reference) will be directly send to the Web-anonymizer. Obviously, this complicates the

implementation of Web-anonymizer considerably which can lead to erroneous services. Furthermore, such anonymizer usually work much slower than proxy-based solutions.

We have carefully studied the above mentioned *Anonymizer* service (cf. [5]). Our analysis revealed that the *Anonymizer* actually anonymizes a lot of privacy-critical data such as the *from*, the *referer* and the *cookie* fields. But other critical data like, for instance, *client-IP* and *cache control* are forwarded unmodified. Furthermore, it is not possible to selectively de-activate the suppression of cookies. Hence, servers which require cookies are not accessible via the *Anonymizer* service. In addition, it should be noticed that the *Anonymizer* does not encrypt the transferred data. Since this restriction is not dictated by the inherent Web-server architecture the level of anonymity implemented by the *Anonymizer* is lower (just level 4) than the one that is reachable in principle. In fact, such anonymizers could provide level 5. Web-server anonymizer could even be improved by using them in a cascading manner which would result in level 6 anonymity.

Crowds

The Crowds-project [8] follows a completely other approach by hiding a user within a crowd. To this end, a user's message is sent encrypted to a randomly selected member of the crowd which selects another receiver among the crowd or sends the message to its final destination. Obviously, a small crowd is not sufficient to guarantee anonymity. The Crowds service suppresses a lot of critical data like *from*, *cache control*, *cookie*, *X-forwarded* and *referer* fields. In addition, it anonymizes the *accept* as well as the *user-agent* data by replacing the original data with default values. But this can cause problems, if a server who interprets this data runs into trouble by using the default values. For example, the value *zip* used to replace other data items in the field *accept-encoding* is not defined in the HTTP specification and might lead to a faulty server action. Furthermore, under Crowds unknown headers possibly containing critical information are transferred without modification. Another disadvantage of the Crowds service is that it is not usable in conjunction with firewalls on the client side. Despite of these problems Crowds is applicable in a cascading manner. Hence, anonymity level 6 is reachable in principle. But it should be noticed that message delivery is considerably delayed through cascades of Crowds servers. Note, that the client of a WWW request is not able to determine the crowd members which are involved in the delivery of his message. That is, unreliable members, untrusted members or nodes that are not online might be randomly selected. The service can be improved, if the clients can select a route depending on information about the current availability and load of crowd members.

2.2. EMAIL AND NEWS

Since email and news are asynchronous services it is not necessary that the messages are delivered to their final destination immediately. This is exploited by remailer services which are the most popular anonymizers in this area. The original idea goes back to D. Chaum [1] who proposed the mix approach to transform messages. A user of a remailer service must explicitly send his messages to the remailer. The remailer removes the header information and forwards the message. To thwart traffic flow attacks, the remailer usually delays message forwarding and puts dummy messages into the message stream. This kind of functionality characterizes the so called type 1 or cypherpunk remailers. A list of available remailers can be found under <http://anon.efga.org/Remailers/TypeIList>. The second class of remailers are the type 2 remailers or mixmaster (<http://anon.efga.org/Remailers/TypeIList/type2.list>). But, using this class of email anonymizer requires a specific email client which encrypts the messages and pads them to a uniform length of usually 30kbyte.

To be able to use remailers in a remailer cascade, the sender must encode a chain of encrypted remailer addresses into his mail. Each remailer removes the entry of the chain which has been encrypted with its public key. The encrypted entry contains the address of the next mailer in the chain. After decrypting its entry a mailer is able to forward the mail correctly. This technique ensure that only the first remailer in the chain knows the identity of the original sender and solely the last remailer in the chain knows the final destination address. Though such a chaining is feasible, in practice remailing services just use one remailer. As a consequence, a remailer sees all the critical data contained within the header data fields as only the message payload is encrypted. Hence, current remailers just offer the anonymizing level 5 whereas the level 6 is achievable by cascading remailers.

3. NEW ANONYMIZING SERVICES

The previous sections showed deficiencies of existing anonymizers. They do not provide anonymizing services that sufficiently anonymize all critical data (in particular, IP-addresses and *accept* fields are omitted) and the step-wise deactivation of anonymizing measures are scarcely supported. In addition, most approaches implement an anonymizing level that is lower than the one that might be achievable. Therefore, in the Anonymous-project [5] we developed and implemented new anonymizing services to overcome these revealed deficiencies.

3.1. LOG-FILE ANONYMIZER

Our log-file anonymizer has been developed for the Apache Web-server. We anonymize log-entries which contain the following information: *hostname*, *identd-name*, *user-name*, *time*, *request line*, *status*, *amount of bytes being sent*, *referer*, *user-agent*. The Apache-server has been configured in such a way that it forwards all these data to our log-file anonymizer. This data is anonymized as shown in table 1.

The resulting log-file is a compromise. Our anonymizer tries to offer an appropriate level of anonymity for its users (i.e. level 3) while preserving enough information for service providers, for instance, to, generate meaningful access statistics. Our anonymizer covers all privacy critical data. We are aware that substituting the hostname by the top-level name can lead to problems if the server requires more precise information to perform specific analysis (e.g. recognizing accesses originating from robots). Therefore, for this entry as well as for other ones we offer the option to configure the anonymizing measures according to individual server needs. Nevertheless, it should be clear that a log-file anonymizer is not our first-choice anonymizing technique, because the user is not able to control and to configure it appropriately.

Data	Modification
hostname	.< top-level-domain> (e.g. de)
identd-name	–
user-name	–
time	[day/month/year:hours : 00:00 zone]
request	no modification
HTTP-version	HTTP/1.0
referer	–
user-agent	anonymizing browser, OS, language

Table 1 Anonymized log-file entries

Example:

The protocol entry

```
sunsystem.in.tum.de
[18/Apr/2000:13:13:27+0200]
"GET /images/logo.gif HTTP/1.1"
"http://www.in.tum.de/"
"Mozilla/4.5[en](Win98;I)"
```

is anonymized to:

```
.de
[18/Apr/2000:13:13:00:00+0200]
"GET /images/logo.gif HTTP/1.0"
-
"Netscape(Windows)"
```

3.2. PROXY ANONYMIZER

The anonymizing proxy receives requests and forwards them to the original proxy server after having concealed and modified the relevant data items (see figure 1). Hence, anonymizing is performed on the user side. But our proxy anonymizer is not bothered with cache management or other management tasks as this is still performed by the original proxy server.

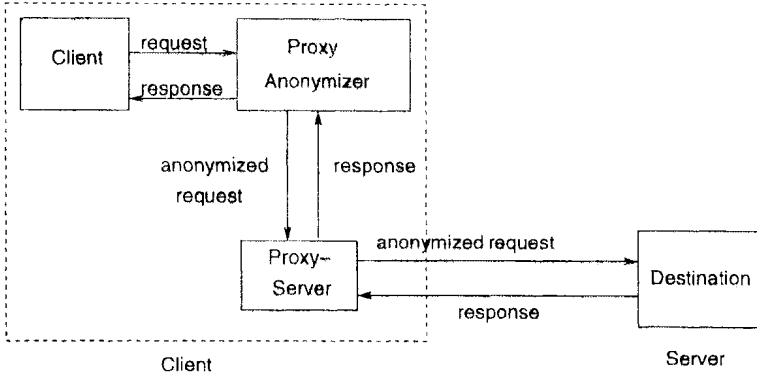


Figure 1 Proxy anonymizer

The anonymizing proxy is registered in the configuration file of the browser. A user can individually configure the anonymizer via a setup page. That is, the user can select the data to be placed into header fields and he can selectively deactivate anonymizing activities for those critical header fields we have previously explained. For instance, the user can determine data items which should be written into the *accept*, *from* or *user-agent* fields. Additionally, he can substitute the data within those fields that contain information about the user's operating system, his hardware platform etc. As a result, requests from different clients are anonymized in a non-uniform manner. Recognizing anonymous requests is therefore difficult for an adversary. Furthermore, our proxy anonymizer allows to configure the data values that the server appends to the fields *cache-info*, *client-IP*, *X-forwarded*, *via*, *forwarded*. Since several servers actually use or require the information contained in the fields *host*, *referer*, *cooky*, *cache-control* and *content-type*, our proxy anonymizer provides the option to selectively activate or deactivate the concealment of these fields. Because of the reasons mentioned in the previous sections, our anonymizer automatically suppresses all unknown header fields. The proxy anonymizer reaches anonymity level 4.

3.3. WEB-ANONYMIZER

To use our Web-anonymizer which is integrated into a Web-server the user must specify the file that should be anonymized and must explicitly call our service via an URL. The data will be anonymized and afterwards forwarded to the specified destination. We just forward the data that is actually required to execute the requested service. For instance, in case of the GET method we will transfer the *host*-header whereas in case of the POST method the *host*-, *content-length*- and *content-type*-headers are forwarded. All server-side answers are anonymized before being delivered to the client (see figure 2). Hence, subsequent accesses of the client on links contained within the answer pages of servers (e.g. encapsulated graphics) are automatically routed via our Web-anonymizer. Besides anonymizing links contained within HTML-data we also anonymize references within JavaScript programs. Furthermore, in contrast to the before mentioned *Anonymizer*, we also conceal FTP references and email as well as news references are automatically forwarded to our anonymizing services (see below). Our Web-anonymizer suppresses all unknown headers and conceals all the privacy critical data discussed in section 1.1. The data can be encrypted by the requesting client, but until now, no cascading anonymizing servers have been implemented yet. Hence, at the time being our anonymizer reaches level 5.

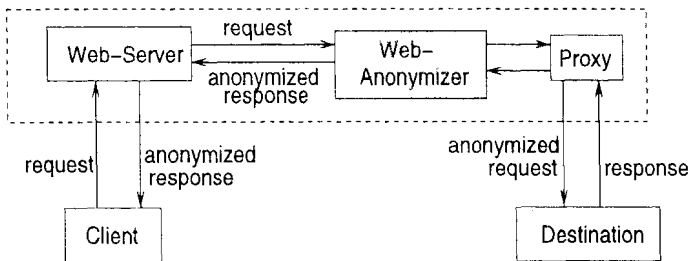


Figure 2 Web-anonymizer

3.4. EMAIL AND NEWS-GATEWAYS

Our email and news anonymizers aim at avoiding the transfer of privacy critical data. To this end, our anonymizing services ask the user to fill in a HTML form with the email data to be anonymized together with the receiver address. Our services just act as gateways as they forward the data to a remailer service within the Internet and send an acknowledgment about the successful forwarding back to the client. The gateways conceal all data except the receiver address, the subject line and the mail payload before sending it to the remailer. A remailer is dynamically selected from the list of remailers. Criteria

to select an appropriate remailer are (1) the availability of the service and (2) the functionality that is supported by the service. For instance, our remailer selection takes into account whether the remailer supports the required message format or news support. The availability of remailers is evaluated based on a remailer statistics that is updated once per hour. The level of anonymity can be increased if the sender uses a SSL connection to our anonymizing service which guarantees anonymity with respect to third parties as well. This level is not achievable by just calling a remailer using an email encryption program like PGP, because these programs solely conceal the mail payload and leave the traffic information unmodified. Since no cascading of our service is provided yet, our anonymizing services reaches level 5.

Our Web-anonymizer, and the email as well as news gateways are available in the Internet see <http://anonymouse.home.pages.de/>.

4. CONCLUSION

With the increasing use of Internet services we are faced with severe threats against user's privacy. All widely-used Internet services and protocols transfer a lot of privacy critical data. By analyzing these data, an adversary is able to generate detailed user profiles. Anonymizing services have been proposed and implemented to thwart these threats against privacy. Log-file anonymizer offer very simple, user-transparent and efficient solutions, but the user can not control the anonymizing activities. Browser-supported proxy servers offer simple and efficient measures as well. Anonymizing services are performed on the client-side and are, hence, controllable. But they are not usable behind a fire-wall and they do not support anonymity against third parties. More elaborated features can be offered by Web-anonymizers. They are usable behind fire-walls and support encrypted data transfer. But Web-anonymizer suffer from slow and complicated anonymizing measures. Emails and news are commonly anonymized by using remailers, but information about the sender is transferred unconcealed to the remailer and an overloaded or not available remailer might cause unacceptable message delays.

Our analysis of available anonymizing services revealed a lot of severe deficiencies. Within the Anonymous project we have implemented new services to overcome the existing problems.

References

- [1] D.L. Chaum. Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms. *Communications of the ACM*, 24(2):84, 1981.
- [2] T. Demuth and A. Rieke. Securing the anonymity of content providers in the world wide web. In *Security and Watermarking of Multimedia Contents*, pages 494–502, San Jose, 1999.

- [3] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. RFC 2068: Hypertext Transfer Protocol — HTTP/1.1, January 1997.
- [4] B. Kantor and P. Lapsley. RFC977: Network news transfer protocol, February 1986.
- [5] A. Pircher. Anonymity in the Internet. Technical Report, TU-München, Master Thesis (in german), August 2000.
- [6] J. Postel. RFC 821: Simple mail transfer protocol, August 1982.
- [7] J. Postel and J. K. Reynolds. RFC 959: File transfer protocol, October 1985.
- [8] Michael K. Reiter and Aviel D. Rubin. Crowds: Anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

Reducing Certificate Revocation Cost using NPKI

Albert Levi and Cetin Kaya Koç

*Oregon State University, Electrical and Computer Engineering Dept., Information Security Lab,
Corvallis, Oregon, USA*

levi@ece.orst.edu koc@ece.orst.edu

Key words: Digital Certificates, Certificate Revocation, Nested Certificates, PKIs, Digital Signatures

Abstract: Problems with certificate revocation status control limit the deployment of Public Key Infrastructure (PKI). Classical certificate paths require revocation control of all certificates on the path. In this paper, we show how the recently proposed NPKI (Nested certificate based PKI) system reduces the number of revocation status controls to at most two. Our analysis also shows that NPKI is not as vulnerable as classical PKI considering the certificate authority compromise.

1. INTRODUCTION

Certificates are the signed objects that bind the cryptographic public keys of the entities to attributes (name, e-mail address, etc.) or to abilities (file access, fund transfer, etc). They are generated by the digital signature of a *Certificate Authority (CA)*. The verifiers use the public key of the CA to verify the certificate content. The system that includes the CAs, end users, certificates and certificate management tools is called *Public Key Infrastructure (PKI)*. Certificates have limited lifespans, but CAs or certificate owners may need to revoke certificates before the expiration time. The reasons of this fact are given below.

- The private key corresponding to the public key in the certificate may be lost or compromised.

- The CA's signature key may be compromised.
- The certification contract may be terminated or the certificate holder's status and abilities described in certificate may change or may be cancelled (as by a person's leaving a job).

Certificate revocation mechanisms must be incorporated into the PKI. The best-known revocation mechanism is the *Certificate Revocation Lists (CRLs)*. A CRL keeps a signed list of the serial numbers of revoked certificates. Usually, the CA is the signer of the CRL for the certificates that it issued. A good discussion on CRLs can be found in [1].

Another practical revocation mechanism is *Online Certificate Status Protocol (OCSP)*, which is published as an RFC [2]. OCSP is a simple request/response protocol that requires online servers, so-called OCSP responder, to distribute the certificate status on demand. Each CA must run its own OCSP responder, unless several CAs unite on this issue.

The literature contains other proposed methods of certificate revocation. Micali [3] proposed the use of on-line/off-line signature scheme for a low-cost check for the "freshness" of a particular certificate. Naor and Nissim [4] proposed authenticated data structures to represent CRLs. Kocher [5] proposed Certificate Revocation Trees (CRTs). CRTs are used to compile the revocation information on a single hash tree. Gasko, Gemmell and MacKenzie [6] proposed EFACTS (Easy Fast Efficient Certification System) that combines the best properties of certificates and CRTs. However, their system is best suited for a single CA issuing large numbers of certificates. Rivest [7] proposed an agent based approach that employs on-line "suicide bureaus" to issue "certificates of health" for certificates. A recent certificate of health must be provided to the recipient along with the actual certificate. A brief taxonomy and overview of certificate revocation methods are given by Myers in [8].

CRLs, CRTs or the on-line revocation systems theoretically may become more centralized by having a single revocation authority to process all revocation data on behalf of CAs. Such an approach has the advantage of gathering all revocation information together, but it creates an extra overhead in terms of messaging among the CAs, certificate holders and the revocation authority. Moreover, several CAs must agree to delegate their revocation responsibility to the revocation authority. Therefore, central revocation authority is not suitable for distributed PKIs where CAs of different organizations interact.

Although there may be some exceptional cases where a single CA issues all certificates in a system, the PKI concept inherently employs a topology of several CAs. Therefore, the verifiers should verify a path of certificates in order to learn the public key of an end user. Consequently, they should check

the revocation status of all certificates on the path. To do so the verifier needs to get the revocation information from all CAs on the certificate path. Thus, the difficulty of certificate revocation is multiplied by the amount of CAs (and certificates) on the path. We stress this problem of “distributed” PKIs that has not been addressed in the literature, except in connection with central revocation authorities that are not suitable for distributed PKIs as discussed above.

Nested certificate based PKI (NPKI) [10] is proposed as a model better suited for distributed applications. It allows rapid certificate path verification. In this paper we analyze certificate revocation rules and advantages of NPKI. NPKI facilitates certificate revocation by requiring revocation status check *only* for the first and the last certificate of a certificate path, no matter how many certificates are on the path. A quick introduction to NPKI is given in Section 2. The certificate revocation rules of NPKI are detailed in Section 3. The implications of these rules and the certificate revocation advantage of NPKI are discussed in Section 4. Section 5 is the conclusions.

2. NPKI

NPKI [10] is based on nested certificates. A nested certificate is defined as a certificate for another certificate. A certificate certified in this way is called a subject certificate. A subject certificate can be a classical certificate or another nested certificate. An NPKI is derived from a PKI with all classical certificates that is shown in Figure 1a. Each CA issues one nested certificate for each certificate issued by its children to form NPKI as shown in Figure 1b. A CA must verify a subject certificate before issuing a nested certificate for it. In NPKI, a nested certificate path (e.g. Figure 2a) is produced for each classical certificate path (e.g. Figure 2b) to verify the certificates of the end users.

The PKI-to-NPKI transition does not change the original PKI topology and trust relationships. This can be seen by examining Figures 1 and 2. The same CAs are in control in both PKI (Figure 1a) and NPKI (Figure 1b). The verifier should trust the same CAs in order to verify the classical certificate path of Figure 2a and the nested certificate path of Figure 2b.

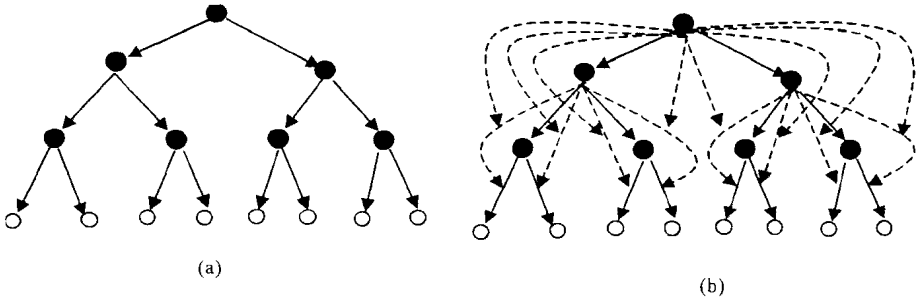


Figure 1. (a) classical PKI, (b) NPKE

- CA/NCA
- End user
- Classical Certificate
- - - - - Nested Certificate

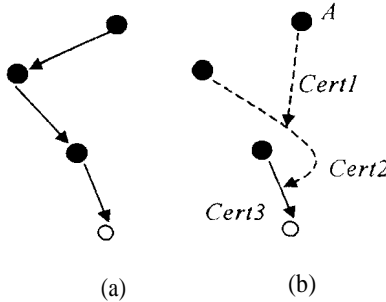


Figure 2. (a) classical certificate path, (b) nested certificate path

The main advantage of NPKE over classical PKI is the improvement in certificate path verification as discussed in [10]. The first nested certificate of a nested certificate path is verified cryptographically. Other certificates, including the last classical one, are verified by hash computations. For example, in Figure 2b *cert1* is verified cryptographically. *cert2* is verified as the subject certificate of *cert1* by only one hash computation. Similarly, *cert3* is verified as the subject certificate of *cert2*. The verifier would need to know only the public key of the first CA (A in Figure 2b). The public keys of other CAs are not necessary for path verification.

3. CERTIFICATE REVOCATION RULES OF NPKI

There are some rules about certificate revocation in NPKI. These rules follow the characteristics of NPKI and nested certificates. This section explains certificate revocation rules. The implications of these rules will be discussed in the next section.

Rule 1: Classical certificates are revocable

The classical certificates for the leaf nodes of NPKI may be revoked, as in classical PKIs, if a necessity discussed on Section 1 arises. The guarantees and bindings given in these certificates are invalidated after revocation.

Rule 2: A revoked classical certificate makes its nested certificate path useless

The ultimate aim of a nested certificate path is to verify the classical certificate at the end. Moreover, a nested certificate can exist on only one nested certificate path. Therefore, when a classical certificate is revoked for some reason, all nested certificates on the nested certificate path towards it automatically become useless. Consequently, these nested certificates need not be revoked.

Rule 3: Do not start a nested certificate path with a revoked nested certificate, but revoked nested certificates can still be used on paths

If the key of a CA is compromised, then the nested certificates issued by it must be revoked, because these nested certificates must no longer be verified using the public key of the CA. However, this does not mean that these nested certificates contain bogus information. If someone else can prove that these nested certificates were created before the key compromise, they can still be verified. This can be proved by finding another nested certificate issued for the revoked nested certificate before the revocation time. The verifier can verify the revoked nested certificate as the subject certificate of another nested certificate. For example, consider the example in Figure 3. Suppose the CA, *A*, has issued a nested certificate, nc_1 , at time t_0 . Later at time $t_1 > t_0$, another CA, *B*, has issued a nested certificate, nc_2 , for nc_1 . At time $t_2 > t_1$, the public key of *A* is compromised and nc_1 is revoked. After t_2 , it is not possible to verify nc_1 using the cryptographic method and the public key of *A*. However, it is still possible to verify nc_1 as the subject certificate of nc_2 , which is still valid since *B* had issued nc_2 at time $t_1 < t_2$, i.e.,

before the revocation of nc_1 . Moreover, B had verified nc_1 before issuing nc_2 and guaranteed the legitimacy of the signature over nc_1 . The revocation of nc_1 at $t_2 > t_1$ does not cause the invalidity of the guarantee given by nc_2 at t_1 .

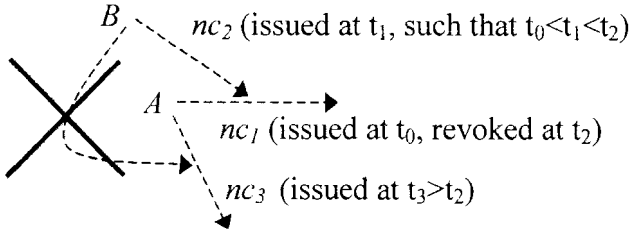


Figure 3. An example case for nested certificate revocation

On the other hand, the counterfeit of A can issue some bogus certificates (for example, nc_3 in Figure 3) at $t_3 > t_2$, i.e., after the compromise of its key. Since B and all other honest CAs are not able to verify nc_3 , they will not issue any nested certificates for it. Thus, the bogus certificates remain isolated and cannot take place on nested certificate paths, as long as they are not verified cryptographically as the first certificate of a path.

Rule 4: No cascaded nested certificate revocations

A revoked nested certificate does not cause its subject certificate to be revoked. A nested certificate does not certify a public key or anything regarding a user. A nested certificate certifies only the relationship of the raw content of its subject certificate and the signature over it. The meaning of nested certificate revocation is that the CA of the nested certificate does not guarantee the correctness of the signature over the subject certificate anymore. However, the signature over the subject certificate can still be verified cryptographically using its issuer's public key. Therefore, nested certificate revocation is not a recursive process towards the end users.

4. DISCUSSION

The above rules imply that the verifier must check the revocation status of two certificates on a nested certificate path regardless of the path length. One of them is the first nested certificate, which is to be verified cryptographically. This certificate must be checked in order not to start the verification process with a bogus certificate (rule 3). Second certificate for which the revocation status must be checked is the last certificate of the

nested certificate path, because it is a classical certificate and the revoked classical certificates cannot be used (rule 1). Other nested certificates on the path need not be checked for revocation, because even if an intermediate nested certificate is revoked this does not cause other certificates to be revoked (rule 4) and it can be used on the path (rule 3).

However, in a certificate path of a classical PKI, all certificates must be checked against revocation. Since all these certificates are from different CAs, different CRLs or OCSP responder contacts would be necessary for the revocation checks. Since there are only two certificate revocation controls in NPKI, the cost of certificate revocation relatively decreases for the paths longer than two certificates as compared to classical PKI.

The revocation status of the first nested certificate of a nested certificate path must be checked since it might have been revoked due to a CA key compromise as discussed in rule 3. This revocation control can be waived if the verifier can make sure about the legitimacy and validity of the public keys that it uses to start the verification process. This would be possible by keeping this CA key information in a local Personal Security Environment (PSE) and by periodically checking the validity of these keys. Similar approaches are proposed by PGP [11] and ICE-TEL [9] systems. However, the revocation status of the classical certificate at the end of a nested certificate path must always be checked.

One can argue that the CA compromise might go undetermined for a long time and during this period some bogus nested certificates can be disseminated. This is still not a big problem and does not require a mass revocation of innocent certificates. Once the breach is detected, it is sufficient to revoke the certificates issued by the compromised CA after the compromise, and the nested certificates issued on them recursively[†]. One may also argue that the counterfeit may change the timestamps in the certificates as if they are issued earlier. This is not correct, because if the counterfeit does so, other CAs realize that something is going wrong and decline to issue nested certificates for the certificates issued by it. Thus, bogus certificates remain isolated.

Above discussion and the rules 3 and 4 also yield that CA compromise in NPKI is not as severe as in classical PKI. The main reason behind this fact is that each CA controls its children by the nested certification process embedded in NPKI. There is no such control in a classical PKI. Once a classical certificate is issued, the issuer can no longer control the activities of the certificate holder.

[†] If this argument is the concern of the system, the verifier should check the revocation status of the first certificate of the path even if he/she makes sure about the validity of the public key of the corresponding CA, because this argument brings out a reason other than CA key compromise to qualify a nested certificate revoked.

Revoked certificates can be kept in Certificate Revocation Lists (CRLs) or handled by other methods cited in Section 1. Each CA manages its own revoked certificates. There may also be nested certificates that are not revoked but are useless (rule 2). This situation inflates the databases/directories. A solution to this problem is to periodically run maintenance programs to locate and delete these useless nested certificates.

5. CONCLUSIONS

Nested certificate based PKI (NPKI) has been recently proposed as an efficient, dynamic and trust-preserving PKI scheme [10]. In this paper we analyzed the revocation characteristics of nested certificates and NPKI. We concluded that it is sufficient to check the revocation status of at most 2 certificates on a nested certificate path, the first and the last certificates, regardless of the number of certificates on the path. The rule for “classic” PKI is to check the revocation status of all certificates on the path, giving NPKI an obvious advantage.

Our analysis also indicates that NPKI CAs are less vulnerable to being compromised than PKI CAs, since their activities are monitored via nested certification.

NPKI does not add any extra burden to facilitate certificate revocation and to make their CAs less vulnerable. These characteristics are the consequences of the nested certification scheme embedded in NPKI.

ACKNOWLEDGMENTS

This work has been supported by rTrust Technologies.

REFERENCES

1. Adams, C., and S. Llyod, *Understanding Public Key Infrastructures*, New Riders Publishing, 1999
2. Myers, M., R. Ankney, A. Malpani, S. Galperin, and C. Adams, X.509 Internet Public Key Infrastructure On-line Certificate Status Protocol – OCSP, RFC 2560, June 1999.
3. Micali, S., Efficient Certificate Revocation, MIT Laboratory for Computer Science, Technical Memo 542b, March 1996.
4. Naor, M., and K. Nissim, “Certificate Revocation and Certificate Update,” *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, pp. 561 – 570, April 2000.

5. Kocher, P., "On Certificate Revocation and Validation," Proceedings of Financial Cryptography 98, LNCS 1465, Springer-Verlag, pp. 172-177, Anguilla, BWI, February 1998.
6. Gassko, I., P. S. Gemmell, and P. MacKenzie, "Efficient and Fresh Certification," Proceedings of Public Key Cryptography (PKC) 2000, LNCS 1751, Springer-Verlag, pp. 342-353, Melbourne, Australia, January 2000.
7. Rivest, R., "Can We Eliminate Certificate Revocation Lists?," Proceedings of Financial Cryptography 98, LNCS 1465, Springer-Verlag, pp. 178-183, Anguilla, BWI, February 1998.
8. Myers, M., "Revocation: Options and Challenges," Proceedings of Financial Cryptography 98, LNCS 1465, Springer-Verlag, pp. 165-171, Anguilla, BWI, February 1998.
9. Chadwick, D. W., A. J. Young, and N. K. Cicovic, "Merging and Extending the PGP and PEM Trust Models – The ICE-TEL Trust Model," *IEEE Network*, vol. 11, no. 3, pp. 16-24, May/June 1997.
10. Levi, A., and M. U. Caglayan, "An Efficient, Dynamic and Trust Preserving Public Key Infrastructure", Proceedings of 2000 IEEE Symposium on Security and Privacy, pp. 203-214, Oakland, CA, USA, May 2000.
11. Zimmermann, P., PGP User's Guide, available with free PGP software from <http://www.pgpi.com>.

This Page Intentionally Left Blank

The Need and Practice of User Authentication and TTP Services in Distributed Health Information Systems

BERND BLOBEL, PETER PHAROW

Institute for Biometrics and Medical Informatics, Otto-von-Guericke University Magdeburg, Leipziger Str. 44, D-39120 Magdeburg, Germany.

E-Mail: Bernd.Blobel@MRZ.Uni-Magdeburg.DE

Key words: Security services; Security mechanisms; Smart cards; Trusted Third Party

Abstract: Shared care requires open distributed information systems for supporting communication and co-operation. Regarding the sensitive character of personal medical information, such communication and co-operation must be provided securely. Meeting the European as well as national legislation, several projects such as ISHTAR, TrustHealth, MEDSEC, EUROMED-ETS, and HARP have been launched by the European Commission for specifying, implementing and evaluating appropriate security solutions. Based on the mentioned projects' results, a trustworthy shared care infrastructure is discussed in the paper.

1. INTRODUCTION

The well-known changes in healthcare like specialisation and decentralisation, the need for efficiency and efficacy, but also the increased mobility of patients and health professionals, the flexibility (working in different application environments) as well as regionalisation or even internationalisation of healthcare cause a paradigm change in health to shared care. Adequate health information systems, which have to be distributed and co-operative must support shared care structures, too. Exchanging personal medical data, communication and co-operation especially in health have to be provided securely.

In Europe, the basic legal issues about security for personal and medical information are ruled in the „European Directive 95/46/EC on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of such Data“ [1] and in the „European Recommendation No. R(96) of the Committee of Ministers to Member States on the Protection of Medical Data“ [2]. Based on results of several projects related to security in healthcare and funded by the European Commission, such as ISHTAR, TrustHealth, MEDSEC, EUROMED-ETS, and HARP, some security solutions for the mentioned type of systems will be discussed in the paper [3, 4].

The care of cancer patients is a long-standing example of shared care. As an integrated clinical cancer documentation system, the Clinical Cancer Register Magdeburg/Saxony-Anhalt has been the first distributed interoperable regional healthcare information system in Germany. The highly sensitive content of the Clinical Cancer Register information and our open system architecture are demanding a high level of security, reliability, and privacy of information records and communication procedures.

2. SECURITY REQUIREMENTS AND SOLUTIONS IN DISTRIBUTED MEDICAL RECORD SYSTEMS

Communication and co-operation between a large number of varying users across the boundaries of domains as departments, organisations, regions, or even countries are increasingly bearing security threats of the personal medical information collected, stored, processed, and communicated in Health Care Establishments (HCEs) [5, 6].

Security is a very complex issue related to legal, social, ethical, physical, organisational, and technological dimensions defined as security policy. In that context, security addresses human, physical, system, network, data, or other aspects. Regarding basic requirements of secure communication¹ and secure co-operation² in distributed systems based on networks, basic security services are required [5, 7]. These services have to provide identification and authentication, integrity, confidentiality, availability, audit, accountability (including non-repudiation), authorisation, and access control. Additionally, infrastructural services such as registration, naming, directory services, certificate handling, or key management are needed. Especially but not only in healthcare, value added services protecting human privacy rights as

¹ communication security consisting of secure connectivity and secure message transfer

² application security

anonymisation or providing accountability as time stamping and registration of professionals are indisputable. The services mentioned could be provided by applications or by external objects. With the growing use of complex middleware architectures such as CORBA, DCOM/ActiveX et al., this functionality will also be served by the implemented middleware. For further details see [5, 8].

The Magdeburg Medical Informatics Department is hosting and maintaining Germany's first health record system in oncology supporting different providers who are involved in cancer patients' care and belong to different organisations within the regional shared care system in oncology. Structure and functions of the Clinical Cancer Register Magdeburg/Saxony-Anhalt are described, e.g., in [9, 10].

The next sections are going to discuss some of the models used, shortly considering the services mentioned.

2.1 Security Services

For analysis and design of secure health information systems, a comprehensive set of models has been developed at beginning of the nineties which is only partially issue of this paper. The approach is based on a generic component paradigm, e.g., published in [11]. This paradigm reflects the different views according to the ISO Reference Model – Open Distributed Processing [12] as the view on the enterprise hosting the system, the view on the information managed, the view on the computational principles, the view on the engineering aspects, and finally the view on the technology used. Regarding the granularity, different levels from concepts through services, up to mechanisms and algorithms can be defined. Such a layered model is shown in figure 1. At the conceptual level, the concepts quality, safety, and security, and regarding the latter the concepts of communication security and application security can be distinguished. The basic service considering communication between principals (users, systems, applications, components, objects, etc.) is the strong mutual authentication of these principals controlling the access to the other principal. Furthermore, the principals' accountability for information communicated as well as its integrity, confidentiality, and availability must be guaranteed. Additionally, notary's services like certified time stamps have to be delivered. Regarding application security services, authorisation and accountability according to the dedicated roles of principals following the rules established in the policy have to be controlled. Furthermore, also access control to information as well as its integrity, confidentiality, and availability must be ensured. Beside notary's functions, the comprehensive and trustworthy audit is essential [5, 7].

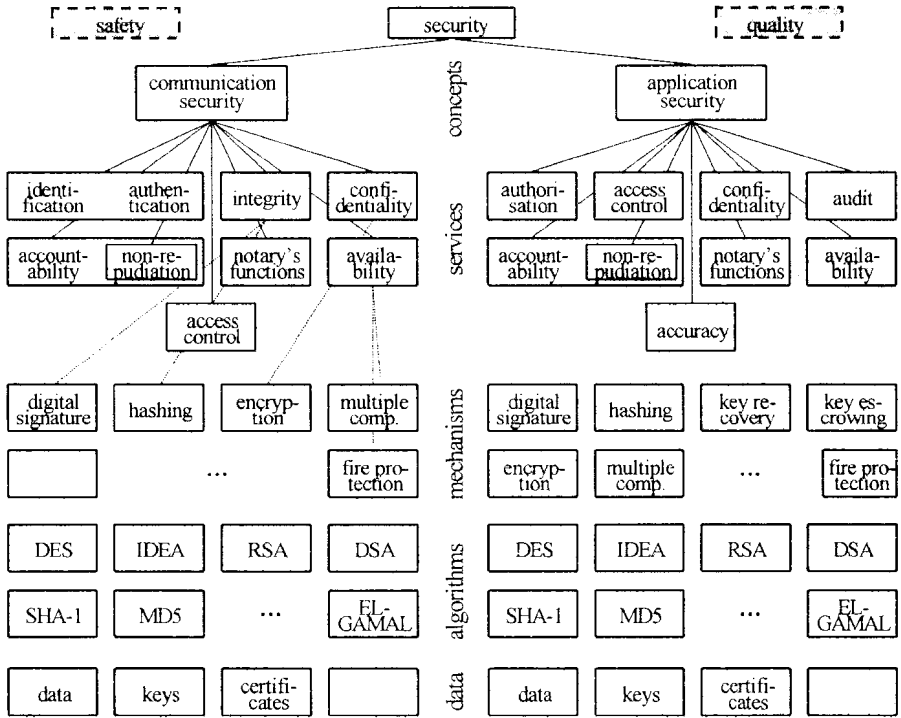


Figure 1. Layered Model of Concepts-Services-Mechanisms-Algorithms-Data Relationship

2.2 Domain Model

As information systems scale to regionally, nationally and even internationally distributed systems, their complexity has to be reduced in order to remain manageable with respect to both security specification and threat model. Collecting similar components into security domains, representing special scope to the system usually does this. Common features allowing grouping are, e.g., organisation, functionality, responsibilities, obligations, technical basis, policy, application domain, or jobs. According to the CORBA Security Model [5], there are three major types of security domains:

- the security policy domain,
- the security environment domain, including message protection domain and identity domain, and
- the security technology domain.

A security policy domain comprises participants and system components that are intended to operate under the same security policy. A security policy is a contiguous strategy of organising security by establishing consistent rules, duties, and liabilities to enforce information security, as well as by defining and controlling authentication, access control, accountability, and others [6, 7]. A *security authority* administers each security policy domain.

A security environment domain is the scope over which the enforcement of the security policy is achieved by means local to the environment, i.e. without any help from other domains. A security environment domain is implementation-specific in the sense that it uses services from the underlying operating systems, basic protection mechanisms and communication services of the lower OSI layers to provide message protection. Therefore, the domain is also called message protection domain. Within a security environment domain, an identity domain can often be defined specifying common access control rules, rights and privileges. Usually, weak authentication procedures are in place (mutual trust of members). A typical example of a security environment domain is a department. In rare cases, where a whole enterprise employs a closed (centralised) system the enterprise as a whole can be regarded as a security environment domain, too.

A security technology domain uses homogeneous technology to enforce a security policy. Given this homogeneity, a department or a whole institution can represent a security technology domain. However, in open distributed systems such homogeneity rarely occurs.

To give a practical example, the purpose of security domains is to form groups of mutual trust defining a special level of risks and therefore demanding a set of countermeasures. Assuming adequate characteristics, departments, enterprises, institutions, and even distributed organisations can be considered as domains. These domains are assumed trusted and trustworthy environments, which must only be protected against external threats. Therefore, special security measures are required only for communication with partners outside the domain and are thus implemented at the domain boundaries. Examples for such advanced security measures are *firewalls*, *proxy servers*, and *external access LANs*. External services like WWW are kept outside the security domain. Bypassing the firewall by, e.g., “private” lines to the outside world using modem-mediated connections without special security measures must be prevented. To avoid unauthorised access, *routers* provide the association of locally external members of the enterprise representing the same security policy domain but different security environment domains. Because of the different security environment and security technology domains, message protection as well as authentication means are often required. To protect sensitive data according to the common view, between different security policy domains the highest

level of security, but within the same security environment domain representing an identity domain (and even the same security technology domain) the lowest level of security, is required.

At least partly, centralised architectures and non-co-operating institutions fulfil the scope of the domains described. They are traditionally considered as closed and therefore secure systems. The trustworthiness of such systems is mainly based on the trust of both technology and involved subjects (users and administrators). Distributed systems are more vulnerable to security breaches than the traditional systems, as there are more places and opportunities that the system can be attacked. Further, we find the more complicated conditions of different domains. Nevertheless, the structural changes in healthcare systems as well as technological developments are demanding the inherent distributed nature of health information systems.

On the other hand, 70 to 95 percent of attacks on information systems are executed by insiders, as could be shown by own investigations performed in Germany as well as by data from the USA [8]. In that context, the following chapters describe future health information systems and related security, assuming open and non-trusted conditions. The shared care approach requires that the reliability of processes and information must be assured by corresponding security-related measures [5].

3. USER RELATED SECURITY SERVICES

Sharing care as well as the resulting communication and co-operation in healthcare have to be person-related. Beside social and human reasons, this is caused by the legally binding property of business processes (including liability issues) with its corresponding security services like authentication and digital signature [3, 4, 6]. In addition, application security services such as authorisation and access control depending on structural or functional roles have to be person-related too. The structural role reflects position and responsibilities within the organisational hierarchy, whereas the functional role reflects the concrete functional and procedural activities in the care environment [5, 7].

Communicable medical information systems need not be bound to networks. Data may be recorded, stored, and processed at other media. In that context, the development of smartcard technologies especially in Europe enables alternatives.

Patient Data Cards (PDC) are smartcard-based medical application systems. Providing patient's informational self-determination as a specific type of user relationship, a PDC requires a special access control management to keep the security level and trustworthy relationship

guaranteed to the patient [1, 2, 5, 9]. Involved in the DIABCARD project [3, 4] of smartcard-based information systems funded by the European Commission and supporting communication and co-operation of diabetes care, the Magdeburg Medical Informatics Department has specified and implemented corresponding user related security services considering both health professionals and patients [13].

An appropriate tool to provide person-related security services bearing information items needed as cryptographic keys and certificates is the use of identity-bound and role-bound tokens. In Europe, the smartcard technology has been preferred as secure and payable solution provided as Electronic Identity Card (EIC) and/or Health Professional Card (HPC), which could also be used in a pan-European Healthcare Network based upon the Internet and its tools [3, 4, 14]. Guaranteeing a bilateral trustworthy patient-doctor relationship, the patient needs such a token like an electronic Patient Identity Card (PIC), too. This PIC could be combined with other functionality as patients' medical data on Patient Data Cards (PDC) or patients' insurance cards. Currently, such PDC with PIC functionality is under implementation as next generation DIABCARD.

Facilitated by several projects funded by the European Commission, the Health Professional Card will be widely used in most of the European countries. This process is supported by governmental laws as, e.g., in France or by common initiatives of the physicians' organisation and other bodies of the physicians' self-government as, e.g., in Germany. To enable communication and co-operation across national borders, architecture and interfaces providing access to the card have been standardised at the European scale as CEN TC 251 prENV 13729 "Health Informatics – Secure User Identification – Strong Authentication using Microprocessor Cards (SEC-ID/CARDS)" [15], which is compatible, e.g., to the German HPC Specification [16]. Also card readers and interfaces to the hardware and software components of the application environment must be agreed on. EC-funded projects such as TrustHealth, CARDLINK, and DIABCARD [3, 4] provided corresponding specifications. The following sections explain the HPC concept and its related TTP infrastructure in some more detail.

4. THE EUROPEAN HEALTH PROFESSIONAL CARD

The cryptographic basis for the HPC security functions' model is an asymmetric algorithm, e.g. RSA or elliptic curves. Therefore, a specific key pair is generated, consisting of a private key (the owner's secret) and a public key. The private key is securely stored in the HPC and does never

leave this environment; the public key is stored in a public directory as part of a public certificate. To enable different security services, three key pairs are required to fulfil the security needs. There is one key pair for authentication procedures, another one for digital signatures, and the third key pair for encryption/decryption of, e.g., session keys. In some specifications, a fourth key pair is requested for encrypted storage of data in databases or electronic archives in order to allow a key-escrowing scheme only for storage keys if needed.

The HPC is further prepared to store additional information about the cardholder's identity, e.g. his or her name and address. Nevertheless, the HPC is a professional smartcard. And as stated before, the care process in general and the related communication and co-operation in healthcare and welfare have strictly to be person-related, considering the liability and the legal binding as well as corresponding security services. Therefore, Public Key (PK) certificates are used. Connected by identification means, the related attribute certificates are dedicated to access control functions [17]. Especially the application security services as, e.g., access control depending on structural or functional roles have to be established in a secure manner. Hereby, the structural role reflects administrative aspects as the position and the related responsibilities within the organisational hierarchy, whereas the functional role reflects the concrete functional and procedural activities in the context of the specific care environment. Currently it is not yet decided whether certificates will be stored only in directories, only in the card or possibly both could be done. If it should be done in the card, a lot of further work has to be done in the area of Card Verifiable Certificates (CVC)

5. THE RELATED TRUSTED THIRD PARTY STRUCTURE

The European TrustHealth project has started to describe the processes within the real world and the electronic world in terms of security services and their service specification [18]. Trusted Third Party (TTP) organisations have to provide different services.

In the traditional world of papers, one will find the authorities responsible for issuing authentic documents of an individual. That includes e.g. a registration office for inland and travel passports and a qualification authentication authority (QAA) for diploma etc. Regarding our movement to eHealth, any kind of information or certain data items are processed and transmitted from the real world into the electronic world by specific interfaces. All authorities of the electronic world are components of a Trusted Third Party structure.

Based on the formerly real world data items mentioned above, and connected to a unique distinguished name (DN) created by a Naming Authority (NA), a Registration Authority (RA) within the electronic world issues authentic documents (paper or database) of identity (Public Key Registration Authority - PK-RA) of profession (Professional Registration Authority - Pr-RA). Besides that, a Key Generation Authority (KGA) generates specific key pairs (see above). This could be done as a centralised process within the TTP (CKG), or it could be done locally within the user's secure environment (LKG). The decision whether it is allowed to generate keys outside a TTP environment is more a political than a technical one.

Authentic links between an individual's DN, his or her authentic ID documents and his or her Public Key are used to issue a Public Key Certificate (PK-Certificate) by a public Key Certification Authority (PK-CA). A Professional Certification Authority (Pr-CA) linking professional information items without any key to issue a Professional Certificate (Prof. Certificate) does the same. All different data items, keys, and related certificates are necessary to establish the security services of identification and authentication, integrity, confidentiality, availability, and accountability. For legal reasons (responsibility) and for reasons of trust (professional bodies), different organisations become responsible for the different steps of the registration and certification processes. Now, how is this rather complicated procedure really performed within the Magdeburg pilot environment?

The University Hospital of Magdeburg (UHM) including its cancer centre on the one hand and the Physicians' Chamber of the German federal state of Saxony-Anhalt (PCSA) on the other are currently authorities of the real world in terms of profession. For identity purposes, the German inland passport issued by an official German registration office is used. Considering current developments, electronic components of the TTP at UHM and at PCSA acting both as NA and Pr-RA have been established which are also applicable as a PK-RA using the individuals' passport for identification. For issuing PK certificates, our German TrustHealth partner GMD Darmstadt (Gesellschaft fuer Mathematik und Datenverarbeitung) provides the services needed. In the future, a CA officially based on the requirements of the new German Digital Signature Law and Act will be introduced. The CA has set up a public directory service including the procedure of Certification Revocation List (CRL). A locally managed directory service as a back up of the CA service is available as long as connections between a health professional and the Magdeburg Registry will occur.

The generation, distribution, and revocation of keys, certificates or even cards as well as the provision of corresponding information services as directory services, often summarised Public Key Infrastructure (PKI),

require an appropriate infrastructure of national or pan-European TTP services.

Within the TrustHealth project mentioned already, the Magdeburg Medical Informatics Department developed, implemented, and evaluated a trustworthy health network for shared care in oncology called ONCONET [10]. As the first one in Germany, the ONCONET is based on standardised tokens and services such as HPC and TTP services. At the same time, the ONCONET has been the first pilot for the German electronic doctor's license [16]. The ONCONET will be presented shortly at the end of the paper.

6. THE PROCEDURE OF HPC DISTRIBUTION

The health professional fills out an application form consisting of several specific registration forms [3, 4] with all details asked for, and gets his distinguished name (DN) by the Naming Authority (NA). The PCSA for all physicians and the UHM (Cancer Centre) for non-physicians verify and "certify" the identity and the professional details as qualification, speciality, role etc. of the health professional by signing the complete registration form. As a Registration Authority (RA), they send the preliminary authentic paper form or the related electronic authentic document to a selected Certification Authority (CA) "by law" which simply means that the CA has to be evaluated by legal authorities in Germany and has thus to be certified as strictly following German electronic signature legislation.

As soon as all the procedures of card issuing and the related TTP services are finalised (the keys are generated, the card is initialised and personalised, the certificates are created, and the directory update is done), the card and the PIN code to just open it are sent to the responsible Registration Authority (RA) using separate ways. PCSA or UHM get the card and the PIN code to deliver both to identified and authenticated users. The health professional can do this identification by providing either inland or travel passport as mentioned above.

Within the RA environment, a simple test application is used to verify card and PIN operations. Therefore, the user can check both the Health Professional Card and the access to it before he or she leaves the office. The user is requested to specify a new PIN after this first use of the HPC because the former PIN is just a so-called "transport PIN". If everything works as properly as expected, the health professional is able and allowed to use his or her HPC for each security functionality within the given pilot environment. The medical background of the Magdeburg cancer documentation

application and the related oncological network will not be described here. This information can be found in [9].

For improved data protection and data security reasons, the further development of smartcards and related authentication mechanisms will lead to the use of biometric algorithms as, e.g., fingertip, eye analysis, or voice analysis. The current European HPC concepts consider this new trend by specifying requirements for those biometric algorithms and describing the needs of related interfaces.

7. INTERNET BASED SECURITY INFRASTRUCTURE

Beside of the network security services mentioned above, several projects funded by the European Commission currently aim the development of a pan-European healthcare network based on the Internet and its WWW tools. In that context, security infrastructures based on standardised hierarchical TTP structures have been installed. They are managing a Public Key infrastructure and the related mechanisms, providing CA services including cross certificates to other TTP hierarchies [3, 4].

Figure 2 shows the general schema of this first distributed international TTP architecture in healthcare developed for another European project called EUROMED. EUROMED-ETS itself has involved the pilot sites University of Athens in Greece (ICCS), University of the Aegean in Greece (UoA), University of Calabria in Italy (UoC) and University Hospital of Magdeburg in Germany (UHM).

Using the example of the Magdeburg UHM part of the solution, figure 3 presents the hierarchical TTP structure of this distributed international healthcare EUROMED-ETS TTP architecture. ICCS at the National Technical University of Athens (NTUA) in Greece hereby represents the root-CA. Below this top-level CA, ICCS has implemented another CA service for the EUROMED-ETS (ETS Consortium) purposes. This CA called EUROMED-ETS-NTUA has been certified by the root-CA and has then certified the Magdeburg CA (UHM CA) located at a specific CA server (cabmi1.medizin.uni-magdeburg.de). Besides the certification of other CAs, the ETS CA has to issue identity certificates for the ETS community, as shown in the example above following the hierarchical scheme leading to a user ID certificate (Peter Pharow's UoA ID).

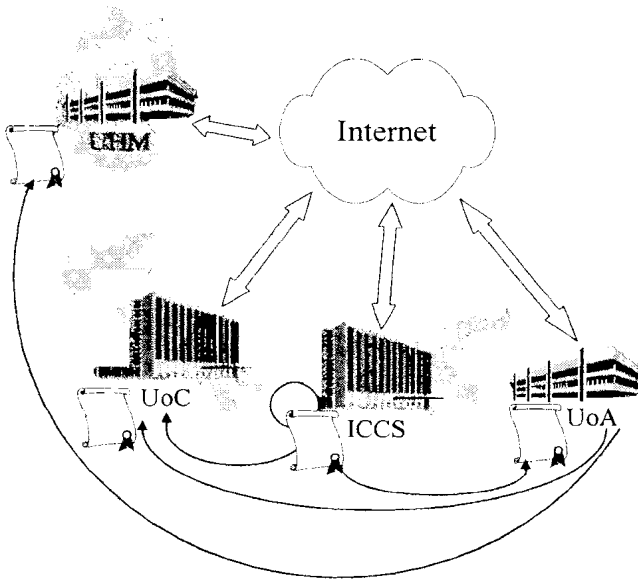


Figure 2. EUROMED-ETS Pilot Architecture for Internet Security Services

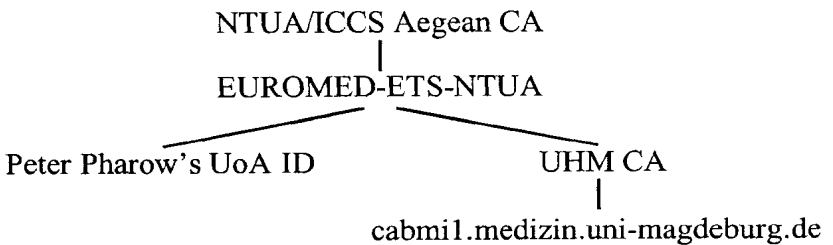


Figure 3. Schema of the Hierarchical TTP Structure

Internet tools as browsers are being completed with enhanced security functionality soon. Important Internet application environments as, e.g., Java have and will further get improved security mechanisms. Additionally, the HPC has been introduced in the Internet-based communication infrastructure mentioned above. Finally, especially security requirements for handling patient's medical and administrative data using the Internet have been mentioned during the IMIA WG4 Working Conferences held in Osaka and Kobe (Japan) in 1997 and in Vancouver (Canada) in 2000 (e.g. [8, 14]).

Following the requirements of the market as well as the European e-Health strategies, the European Commission has agreed to further investigate Internet and security issues. Started in January 2000, a project called “HARP – Harmonisation for the security of the web technologies and applications” is currently focusing on secure medical applications accessible via Internet [3, 4]. Based on former investigation especially in the context of traditional TTP services such as card generation and certificate issuing for human beings, HARP is dealing with a more flexible strategy concerning also systems, documents, applets, etc. as part of a security infrastructure thus allowing them to authenticate themselves towards other *principals* and to e.g. sign transmitted data.

The overall objective of the HARP project is the development of new technologies and tools for the integration of Web-oriented security systems and the combination of coherent services to demonstrate and quantify the value of security tools/mechanisms/systems harmonisation in business and citizen needs in the Information Society. This overall objective is broken down into the following sub-objectives:

- a) Review/analyse Web components used in the telemedicine sector in terms of security;
- b) Investigate the impact of TTPs in the security of Web-based telemedicine applications;
- c) Develop harmonising software and tools to cope with the diversity of the Web components;
- d) Design, integrate, validate a harmonising, cost-effective, user-friendly security platform based on TTPs for securing integrated telemedicine applications;
- e) Demonstrate HARP's integrated security solution in the telemedicine sector;
- f) Disseminate the project results to the widest possible audience.

To achieve the project's objectives the work is split into four phases. In phase A (“Feasibility Study”), HARP has already adopted and newly developed metrics, methods, criteria and test methodologies. These means have been used to identify, classify, evaluate and compare Web components, to investigate how TTP technology can be used to prevent the various risks introduced by the Web use, and to draw evaluation criteria for the project results and pilot operation targeted in the telemedical sector.

As an outcome of phase A, the HARP consortium decided to follow both server-centric and user-centric approaches to introduce a security infrastructure over the open Internet that is prepared to allow secure access to, and secure download of, documents, guidelines, application form, software applets, etc. After all, this strategy will allow HARP to offer both products and services.

In phase B (“Design and Development”) that has started recently, harmonising tools and mechanisms are to be designed so as to allow TTPs to cope with the diversity on the Web-based telemedical applications. A cross-security platform based on the TTP technology will be introduced soon. Platform-specific security features will be isolated and communicate with them through an abstraction layer that will work for all platforms. This will be accomplished by letting visible interface of a platform specific case define how client code accesses a function without regard of how the function is implemented.

In phase C (“Pilot Evaluation”), the designed platform and the developed TTP services/functions will then be integrated and evaluated by medical users (hospitals). For the evaluation, phase A will be used as a yardstick. The trial network will reflect the TTP architecture in specific telemedicine scenarios designed already.

Finally, phase D (“Promotion”) includes the production of guidelines that will cover all the information cases, techniques and algorithms. Workshops and meetings with key actors from health authorities, industry, business and academia will help defining security specifications and conditions for commercial deployment of related products. HARP will establish a continuous collection and dissemination of results obtained in security projects.

8. THE ONCONET SAXONY-ANHALT

Within the European TrustHealth project, a German demonstrator based on the solutions illuminated has been established presenting a comprehensive security infrastructure for health information systems. Supporting communication and co-operations between HCEs dealing with cancer patients’ care, the healthcare network demonstrator is called ONCONET. Using HPCs and TTP services at least partially provided by the Physician’s Chamber of the federal state Saxony-Anhalt, the network enables communications between health professionals as well as between them and the Clinical Cancer Registry Magdeburg/Saxony-Anhalt which is hosted at the Magdeburg Medical Informatics Department. It allows the trustworthy exchange of doctor’s reports but also any type of file (HL7 messages, images). Furthermore, pre-defined or even free SQL (Structured Query Language) queries are possible. For more detailed information about the ONCONET solution see, e.g., [10].

9. CONCLUSIONS

Meeting the shared care paradigm, future health information systems will be distributed, interoperable and Internet-based. Because such health networks deal with personal medical data, information systems must run in a trustworthy way. Within its research and development programmes, the European Commission launched a set of projects for specifying, implementing, and evaluating advanced solution for security services in health information systems. Exploiting the results of different projects, the first German distributed secure health network and electronic medical record system has been implemented. In that context, the standardised European Health Professional Card has been combined with Trusted Third Party services which are currently under enhancement within the HARP project. Including security solutions for smartcard-based medical information systems held by the patient as Patient Data Cards, a comprehensive security framework for health could be provided first in Europe.

10. ACKNOWLEDGEMENT

The authors are indebted to the European Commission for funding as well as to the partners of the projects ISHTAR, TrustHealth, EUROMED-ETS, MEDSEC, and HARP. Furthermore, they would like to thank the colleagues of the Physician Chamber Saxony-Anhalt as well as of the health care establishments involved in the ONCONET for their engagement.

11. REFERENCES

1. CE: Directive 95/46/EC on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free Movement of such Data. Strasbourg 1995.
2. CM: European Recommendation No. R(97) of the Committee of Ministers to Member States on the Protection of Medical Data. Strasbourg 1997.
3. European Commission: Projects of the Fourth EU Health Telematics Applications Programme. <http://www.cho.be/projects/>
4. IBMI Projects. <http://www.med.uni-magdeburg.de/fine/institute/ibmi/dmi/respro.htm>
5. Blobel B, Baum-Waidner B: Current Security Issues Faced by Health Care Establishments and Resulting Requirements for a Secure Health information System Architecture. In: The ISHTAR Consortium (Edr.): Implementing Secure Healthcare

- Telematics Applications in Europe, pp. 101-147. Studies in Health Technology and Informatics, Vol. 66. IOS Press Amsterdam 2001.
6. The SEISMED Consortium (Edr.): Data Security for Health Care. Volume I-III. Studies in Health Technology and Informatics, Vol. 31-33. IOS Press Amsterdam 1996.
 7. Blobel B, Roger-France F: A Systematic Approach for Secure Health Information Systems. Accepted in Int. J. Med. Inf..
 8. Blobel B, Katsikas SK: Patient data and the Internet - security issues. Chairpersons' introduction. International Journal of Medical Informatics 49 (1998), pp. S5-S8
 9. Blobel B: Clinical Record Systems in Oncology. Experiences and Developments on Cancer Registries in Eastern Germany. In: Anderson R (Edr.): Personal Medical Information - Security, Engineering, and Ethics, pp 39-56. Springer, Berlin 1997.
 10. Blobel B: Onconet: A Secure Infrastructure to Improve Cancer Patients' Care. European Journal of Medical Research (2000) 5: 360-368.
 11. Blobel B: Application of the Component Paradigm for Analysis and Design of Advanced Health System Architectures. Int. J. Med. Inf. 60 (2000) 281-301.
 12. ISO/IEC 10746-2: Information Technology – Open Distributed Processing – Reference Model: Part 2: Foundations
 13. Blobel B, Spiegel V, Pharow P, Engel K, Engelbrecht R: Secure Interoperability of Patient Data Cards in Health Networks. In: Hasman A, Blobel B, Dudeck J, Engelbrecht R, Gell G, Prokosch H-U (Eds.): Medical Infobahn for Europe, pp 1059-1068. Series in Health Technology and Informatics Vol. 77. IOS Press, Amsterdam 2000.
 14. Katsikas SK, Spinellis DD, Iliadis, J, Blobel B: Using Trusted Third Parties for Secure Telemedical Applications over the WWW: The EUROMED-ETS Approach. Int. J. Med. Inf. 49 (1998) pp. 59-68.
 15. CEN TC 251: prENV 13729: Health Informatics - Secure User Identification – Strong Authentication using Microprocessor Cards (SEC-ID/CARDS), Brussels 1999.
 16. The German HPC Specification for an electronic doctor's licence. Version 0.81, February 1999. <http://www.hpc-protocol.de>
 17. Wohlmacher P, Pharow P: Applications in Health Care using Public-Key Certificates and Attribute Certificates. In: Proceedings 16th Annual Computer Security Applications Conference. IEEE Computer Society, Los Alamitos 2000.
 18. Blobel B, Pharow P: Experiences with Health Professional Cards and Trusted Third Party Services Providing Security in Distributed Electronic Records in Oncology. Proceedings of the Conference „Toward An Electronic Health Record Europe '97“, pp 29-39. 20-23 October 1997 London.

6

Is the Performance of Smart Card Cryptographic Functions the Real Bottleneck?

Konstantinos Markantonakis¹

Visa International EU

Access Channels & Platforms

Virtual Visa, PO Box 253

London W8 5TE

United Kingdom

markantk@visa.com

Key words: Multi-Application, Smart cards, Java Cards, Terminal APIs, Performance Measurements, Cryptographic Algorithms.

Abstract: It is generally believed that among the major delaying factors of smart card performance is the speed of the cryptographic algorithms. This is only partially true, as a number of other factors that add substantial delays to the overall performance of a smart card application should also be taken into account. In this paper we analyse the significance of these delaying factors. Furthermore, we also present some performance measurements of the two most widely used terminal application programming interfaces (APIs) and Java Cards. The aim of this work is to emphasise, both to smart card application developers and smart card technology researchers, the importance of these delaying factors and also to provide a reference point as to the performance of each API.

1. INTRODUCTION

Among the major tasks of a smart card application developer is the identification of any delaying factors that slow down the execution of a smart card application. Delays can be encountered either in the application

¹The views expressed are personal to the author and do not necessarily represent the views of any other person or organisation for whom the author works or has worked.

running in the smart card or in the application residing in the smart card terminal, i.e. the client or terminal application.

In the past, with multi function smart cards [1,2,3] the situation was simplified. For example, the performance of a smart card application could be measured with relevant precision, since both the smart card and terminal Application Programming Interfaces (APIs) were architecturally simple. Therefore, the only way to achieve better execution times was application code optimisation.

In the recent years with the introduction of multi-application smart cards [4,5,7] the situation changed. Smart cards became capable of securely hosting multiple applications along with dynamically, securely downloading and deleting applications. As a result, the complexity of the smart card operating system (SCOS) increased exponentially [6,7]. Similarly, the complexity of the terminal applications increased significantly as new architectures [8,9,26] emerged. These technologies aim to offer interoperability between smart cards and card acceptance devices. Moreover, they also hide the details of the underlying terminal operating system. Even at this stage, in order to improve smart card application execution, a lot of effort was still placed in code optimisation, improved smart card virtual machines and providing faster smart card microprocessors.

On the other hand, it is generally believed throughout the crypto communities that smart cards are “anaemic” devices that should do as little cryptographic computation as possible. This view resulted in a race to improve the performance of smart card cryptographic algorithms. Obviously, this approach is by no means wrong but if we look at the problem from a different angle there are also other factors, which if improved will significantly reduce the overall execution time of a smart card application.

We believe that smart card application delays mainly come from sending and receiving data packets to/from the smart card. Although this observation is generally recognised as valid it has not yet received the necessary attention. Furthermore, in various smart card related newsgroups, discussion forums and research papers, questions such as how long it takes to communicate with the smart card or which communication API performs better in terms of speed, are always a favoured topic.

In this paper we attempt to provide some meaningful answers to the above questions. In order to achieve our aim we present some performance comparisons between the two most widely used terminal APIs, namely Personal Computer/Smart Card Specification (PC/SC) [8] and OpenCard Framework (OCF) [9]. Therefore, this paper serves two purposes: First it provides some reference points towards which of the two smart card terminal APIs performs better in the available smart card testing platforms. The results of this paper could also be considered as a reference point when

designing smart card terminal applications. Second it highlights the fact that in order to achieve better smart card application execution times it is important to look into other factors apart from cryptography.

The remainder of this paper is organised as follow. First, we outline the characteristics of how communication is achieved with a smart card, both at the physical layer and higher at the application level. Subsequently, we present the implementation environment and a short but detailed analysis of the design characteristics. Moving to the core idea of this paper we then analyse the results from the test implementations. Finally, we discuss several practical issues that imposed certain design decisions and introduce new concepts to act as directions for further research.

2. BACKGROUND

In this section we provide an introduction on how low and high level communication is achieved between the terminal and the smart card application. We also provide some typical smart card cryptographic algorithm performance measurements.

2.1 Physical Data Transmission to the Smart card

Currently there is only a single channel for communication between a smart card and a terminal. This implies that the card and terminal can only transmit in turn and the other party should be in receiving mode. This operation is known as half-duplex operation. Most smart card microprocessors have a single I/O port but since the ISO standards [10,11] reserved two of the eight smart card contacts for future use, full duplex could become technically feasible.

Communication between the smart card and the terminal takes place serially. This implies that each byte to be transmitted in the communication channel should be converted into eight individual bits that are sent one after the other. Since the data transmission proceeds asynchronously, each byte must also be provided with additional synchronisation bits i.e. a start bit, a parity bit and two synchronisation bits.

The data transmission rate is directly proportional to the applied clock of the microprocessor. This implies that the duration of a data bit cannot be given in absolute terms. However the existence of awkward divider values along with the most common clock frequencies aim to provide a transmission speed of exactly 9600bits/s.

Two of the most common data transmission protocols [11,12] are T=0 and T=1. T=0 is asynchronous, half-duplex, byte oriented, was used in

France during the initial phase of the Smart Card development. It is also used in the GSM smart cards and is more commonly used in most current smart cards.

T=1 is asynchronous, half-duplex, block oriented and was introduced in 1992 as an ISO/IEC 7816-3, Amendment 1 standard. The block is the smallest data unit that can be transmitted. This protocol allows chaining of blocks of data i.e. an arbitrary large block of data may be transferred as the result of a single command by the transmission of the appropriate number of frames chained in sequence.

2.2 Communicating Through a Terminal Application

As previously mentioned, smart card application programming interfaces form one part of smart card technology. Another important aspect is the APIs that allow terminal applications to communicate with smart card applications.

Until recently there were no card reader independent application programming interfaces. Two specific reasons for this are: Firstly, the smart cards and the card reader devices were very closely coupled; there was no need for a card to be used with a different card reader and vice-versa. Secondly, card reader programming interfaces were not standardised, whereas smart card interfaces were standardised.

Thus, the most common method employed when smart card programmers wanted to communicate with a smart card application via a smart card reader was the following: obtain the specific drivers for the smart card reader, install them in the system and subsequently integrate them within the terminal application.

PC/SC [8] was developed by Microsoft, Hewlett-Packard, Siemens-Nixdorf, and smart card manufacturers. PC/SC is tied to the Windows platform and terminal applications can be developed in Visual Basic and various C++ compilers. Currently, most smart card manufactures provide PC/SC drivers for their smart card readers.

Another more recent initiative is the OCF [9], which enables Java applications to communicate with the smart card in a transparent and portable fashion. OCF is written in Java and was primarily developed by IBM and other computer technology providers. OCF permits the client applications to access the smart card irrespective of the host operating system and CAD (Card Acceptance Device or Card Terminal).

2.3 Typical Performance Figures of Cryptographic Algorithms in Smart cards

The following table provides some typical figures [23] for the performance of certain cryptographic algorithms in some typical smart card microprocessors.

Table 1. Smart card cryptographic algorithm timings.

Micro-Processors		ST16- CF54B	ST19- KF16	P83W8516/ 8532	SLE44C- R80S
Clock Frequency		5 MHz	10 MHz	5 MHz	5 MHz
Algorithm	Length				
DES	64 bits	10ms	N/A	10 ms /	3.7 ms
SHA	512 bits	15.2 ms	8.2 ms	5 ms	5.6 ms
RSA 512	Sign with CRT	142 ms	20 ms	37 ms	60 ms
RSA 512	Sign without CRT	389 ms	55 ms	93 ms	220 ms
RSA 1024	Sign with CRT	800 ms	110 ms	160 ms	450 ms
RSA 1024	Sign without CRT	N/A	380 ms	400 ms	N/A
DSA 1024	Sign	N/A	100 ms	150 ms	N/A
DSA 1024	Verify	N/A	160 ms	225 ms	N/A

Please note that these are indicative figures. The implementation of the cryptographic algorithms is based on a specific Gemplus implementation [23]. These figures will be used later on when comparing the performance of the terminal APIs with the performance of certain smart card cryptographic functions.

3. IMPLEMENTATION ENVIRONMENT AND ANALYSIS

Software solutions that use smart cards are separated into the smart card application and the terminal application. In our case, in order to perform the tests, these two distinct entities had to be developed. In this section, we outline the characteristics of these two entities.

3.1 The Smart card Application Development Tools

For the smart card applications we used two of the most popular Java Card API Ver. 2.0 [13, 14, 15] compliant implementations, the GemXpresso

Java Card [16] from Gemplus and the Sm@rtCafé Professional Java Card [17] from Giesecke & Devrient. Each of the development kits came with its own smart card reader and development tools. The GemXpresso card came with the GCR410-X reader and the Sm@rtCafé with the Towitoko PCT-200 reader.

3.2 The Smart card Application

The smart card application receives certain commands from the terminal application and responds accordingly. Initially it checks whether the Application Protocol Data Unit (APDU) [18] contains any data and whether it requests any data to be sent back by the card. This is actually achieved by checking the "Lc" and "Le" parameters of the APDU respectively. Therefore, there are four basic functions implemented by the smart card application:

- The first receives no data from the terminal and sends an ISO exception (2 bytes) back, i.e. "*Sent_0_Get_0*".
- The second function receives no data from the terminal but at the same time the terminal requests some data, (X) bytes from the card, i.e. "*Sent_0_Gett_X*". In order for the data to be sent back to the terminal a "for loop" statement is implemented within the smart card application. Please note that there is some processing overhead at the card side in order to execute the "for loop" statement
- The third function receives (X) bytes from the terminal and sends an ISO exception (2 bytes) back to the terminal. This function will be referred as "*Sent_X_Get_0*".
- Finally, the card receives (X) bytes from the terminal and also sends (X) bytes back to the terminal, i.e. "*Sent_X_Get_X*". Please note that this function runs through each byte provided by the terminal and adds the value of one. The end result is an array of bytes of the same size as the original one but its values are increased by one. Therefore, this function also contains some smart card application overhead (e.g. for loop, addition).

In general, the smart card application contains an APDU dispatcher that will verify the APDU sent by the terminal. The Java source code for the smart card applications is around 5-6Kbytes. The actual smart card application files downloaded in the cards are 1.2-1.9Kbytes. The above functionality is implemented as smart card applications both in the GemXpresso and Sm@rtCafé smart cards.

3.3 The Testing Environment

For the implementation and the testing of the terminal and smart card application we used the following configuration: an Intel Pentium II 400Mhz PC with 128 Mbytes of RAM under Windows NT. We also used the Microsoft Visual J++ Compiler Version 1.02.7318 and Microsoft Java Virtual Machine Ver 5.00.3182.

In order to obtain a meaningful set of results we performed a number of tests. It is important to note that we are running each test in two different smart cards (Sm@rtCafé, GemXpresso) and each card is tested in two different smart card readers (GCR410, PCT-200). We have also developed two sets of terminal applications one for PC/SC and one for OCF as described in the next section

3.4 PC/SC Application Design

PC/SC is enabled when installing the PC/SC base components from Microsoft. Subsequently, the PC/SC drivers for the corresponding smart card readers have to be installed.

For the GCR-410 reader we used the GrSerial Ver. 1.2.11.0 driver downloaded from the Gemplus web page [19], and for the PCT-200 reader we used the Ver 2.14.11 driver downloaded from the Towitoko Web site [20]. The Towitoko (PCT) PC/SC driver does not occupy a COM port from boot time on, and thus it is possible to use any other device after disconnecting the reader. Strangely, the GrSerial (GCR) constantly occupies the COM and as a result if the port is to be used by another application, e.g. OCF, then the device driver should be stopped from the "Devices" menu under "Control Panel" in Windows. In any case it is suggested that the whole "Smart Card Resource Manager" service should also be stopped under the "Services" menu in the "Control Panel" of Windows.

In order to provide a common testing platform between PC/SC and OCF the PC/SC terminal application had to be developed in Java. Up to recently it was impossible to find any PC/SC Java source code samples, even from the Microsoft MSDN libraries. This was the main reason that forced us to create some Java source code wrappers, by using Microsoft J++ Ver. 6.0, for the PC/SC COM service provider's [21]. Eventually, that enabled us to gain access to the PC/SC COM components through Java code.

An interesting observation is the following: initially the GrSerial driver could not work with the Sm@rtCafé smart card. After reporting our findings to Gemplus we were told that, the Gcr410 reader does not relay the TCK byte of the ATR to the driver if the card supports the T=0 protocol. They also stated that this was due to a change in the standards. This was also the

case for the drivers of GemPC240 (Gcr240), and GemPC400 (Gpr400), except under W2K. The OCF driver for the same reader does not check the TCK byte, which explains why the OCF driver worked even with the Gcr410 v1.00. Finally, Gemplus's response was efficient as we were provided with a more recent version of the GrSerial driver that corrected the problem and enabled us to continue our tests.

3.5 OCF Application Design

For the GCR-410 reader we used the Gemplus Card Terminal Ver. 3.0 downloaded from the Gemplus Web page [19], and for the PCT-200 reader we used the Giesecke and Devrient Card Terminal Ver. 1.1 driver obtained through the mailing list of the OCF newsgroup. In order to maintain compatibility between the two testing platforms we used the generic PassThruCardServiceFactory service [9] of OCF.

One of the great advantages of OCF is that it does not constantly occupy the serial port of the smart card reader. This means that it is easy to monitor, by running a serial port-monitoring tool [22], the communications on the serial port. For PC/SC on the other hand the serial port-monitoring program has to be started before starting the PC/SC Resource Manager and subsequently executing the client application and obtaining any results.

4. PERFORMANCE EVALUATION

In this section we compare the performance of OCF and PC/SC for each smart card reader and smart card.

4.1 PC/SC and OCF Results and Performance Evaluation

Different results, i.e. the time in milliseconds to complete the specified task, were generated depending on the actual functions described in §3.2. In addition to the above functions we also provide the performance measurements for connecting to the smart card reader, selecting the smart card application and disconnecting from the reader. We provide the Standard Deviation and Average figures, for each function, based on a total of ten measurements. Please note that all the results are based on the specific configurations. This implies that when referring to comparisons between cards, readers and architectures any general comments are based on the specific versions of the reader drivers, and the specific design of the smart card applications.

Table 2. The performance of PC/SC and OCF on the GCR410 reader.

API/Reader	PC/SC on GCR410				OCF on GCR410			
Smart Card	Sm@rtCafé		GemXpresso		Sm@rtCafé		GemXpresso	
	ST.DEV	AVER	ST.DEV	AVER	ST.DEV	AVER	ST.DEV	AVER
Connect	0.0	0.0	4.8	3.0	8.6	3618.2	10.5	3623.2
Select	0.0	40.0	4.2	52.0	3.2	71.0	5.4	85.1
Send_0_Get_0	4.6	122.4	4.8	147.0	3.3	139.2	3.4	109.5
Send_10_Get_0	5.7	141.1	5.2	166.3	3.2	149.0	5.0	177.2
Send_20_Get_0	4.8	157.2	5.1	185.4	3.1	151.2	0.5	190.3
Send_30_Get_0	5.2	165.3	4.9	203.3	3.1	171.3	0.0	210.0
Send_40_Get_0	4.6	183.4	4.7	220.2	0.5	190.4	0.5	290.7
Send_10_Get_10	6.2	198.1	5.0	174.2	0.0	210.0	4.1	182.3
Send_20_Get_20	3.0	309.5	3.0	271.6	0.5	320.5	2.9	249.3
Send_30_Get_30	7.3	421.5	5.1	324.3	5.0	444.5	0.5	280.4
Send_40_Get_40	5.7	529.9	5.6	366.7	0.0	561.0	4.1	392.3
Send_0_Get_10	8.2	130.1	3.2	129.0	5.0	133.4	0.5	130.4
Send_0_Get_20	7.0	195.3	3.1	201.3	4.3	208.2	5.8	219.3
Send_0_Get_30	3.4	259.5	3.1	231.4	0.4	270.2	8.1	237.3
Send_0_Get_40	5.3	326.2	4.9	253.3	0.4	330.8	5.2	264.5
Disconnect	0.0	0.0	0.0	0.0	0.0	30.0	0.0	30.0
Overall	70.8	3179.5	66.7	2929.0	40.6	6998.9	56.4	6671.8

When comparing the performance of PC/SC and OCF for the Sm@rtCafé implementation and the GCR reader it appears that overall PC/SC is 18.6% faster than OCF. Please note that this figure takes into account the average timings from all functions. When the dependency of the comparatively slow "Connect" and "Disconnect" figures of OCF are completely eliminated, as a potential improvement, then for the Sm@rtCafé implementation PC/SC will maintain, on average, a 7% lead over OCF.

For the GemXpresso implementation on the GCR reader it appears that on average PC/SC is 15,2% faster than OCF. Similarly, if the dependency of the "Connect" and "Disconnect" figures are not taken into consideration, PC/SC maintains on average a 3.1% lead.

As we observe from table 2 there is an obvious lead in the performance of PC/SC over OCF for each individual function in the Sm@rtCafé implementation. For a few functions in the GemXpresso implementation, OCF performs significantly better than PC/SC.

Table 3. The performance of PC/SC and OCF on the PCT200 reader.

API/Reader	PC/SC on PCT200				OCF on PCT200			
Smart Card	Sm@rtCafé		GemXpresso		Sm@rtCafé		GemXpresso	
	ST.DEV	AVER	ST.DEV	AVER	ST.DEV	AVER	ST.DEV	AVER
Connect	0.0	0.0	0.0	0.0	35.0	5080.3	45.9	4096.0
Select	4.7	70.0	4.9	77.1	5.2	114.0	3.1	121.2
Send_0_Get_0	4.0	128.1	4.9	163.2	3.5	131.2	0.5	100.3
Send_10_Get_0	4.9	153.3	5.3	186.1	3.3	159.4	0.0	190.0
Send_20_Get_0	5.0	167.2	5.1	336.7	3.1	161.2	7.8	262.4
Send_30_Get_0	5.3	174.1	0.5	350.3	4.3	178.2	4.7	273.6
Send_40_Get_0	5.4	194.3	0.5	360.7	4.2	192.1	8.4	287.3
Send_10_Get_10	5.3	214.6	4.1	182.2	3.1	221.4	4.7	197.5
Send_20_Get_20	4.6	433.3	3.4	291.4	5.1	326.4	5.5	295.2
Send_30_Get_30	6.5	460.7	4.2	338.5	4.1	438.8	5.8	341.5
Send_40_Get_40	0.3	580.9	5.2	375.5	4.3	552.8	0.5	380.5
Send_0_Get_10	4.9	137.1	4.3	132.4	5.2	1728.3	0.5	10054.3
Send_0_Get_20	0.5	190.3	0.4	220.2	0.5	1772.7	18.3	10063.7
Send_0_Get_30	4.5	300.6	0.5	240.4	0.5	1832.6	4.6	10066.3
Send_0_Get_40	4.1	322.3	4.5	268.4	0.5	1872.5	3.0	10075.6
Disconnect	0.0	0.0	0.0	0.0	8.5	65.0	8.2	63.0
Overall	60.0	3526.8	47.9	3523.1	90.5	14826.9	121.5	46868.4

When comparing the performance of PC/SC and OCF on the PCT reader, i.e. Table 3, the situation becomes more complicated. A closer observation will reveal that the performance of both implementations (Sm@rtCafé and GemXpresso) under OCF is influenced by the extremely slow performance of the "Connect" and "Send_0_Get_X" functions. Specifically for the "Send_0_Get_X" functions, the results are unreasonable and indicate that probably these operations are not handled properly from within the OCF driver of the PCT reader. Therefore, for the sake of completeness and clarity we decided to include the performance of the "Send_0_Get_X" functions in Table 3, but do not take them into account when reaching into certain conclusions.

The performance of PC/SC on the PCT reader for the Sm@rtCafé implementation is on average 16.9% faster compared with the one in OCF. But, this is heavily influenced by the large "Connect" and "Disconnect" figures of the OCF implementation. If the influence of these two functions is removed then PC/SC maintains a marginal lead of 0.2%.

Similarly, the performance of PC/SC for the GemXpresso implementation on the PCT reader is on average 8.8% faster than the

corresponding of OCF. An interesting observation is that when the influence of the “Connect” and “Disconnect” figures of the OCF implementation are eliminated then OCF gains a 9.5% lead over PC/SC.

Another interesting observation, by looking in table 2, is that OCF on the GCR reader demonstrates an overall lower standard deviation, for both smart card implementations, when compared with the corresponding one of PC/SC. This implies that OCF appears to be more stable and produces fewer variations in the measurements. For OCF on the PCT reader, i.e. table 3, the situation is exactly the opposite as the standard deviations are significantly larger when compared with the corresponding ones from PC/SC. The latter observation should be considered of minor importance when taking into account the unreasonable performance of OCF on the PCT reader.

From the figures in both tables we can see that the “Connect” and “Disconnect” figures for OCF are relatively large compared with the corresponding of PC/SC. These delays can be possibly explained on the design of OCF. An interesting observation is that when OCF attempts to establish connection with the reader there is increased hard disk activity as OCF searches for certain Java classes. Therefore, carefully setting the classpath of the testing platform will potentially result in small performance improvements. The slow connect and disconnect figures can be possibly explained by the fact that OCF does not constantly occupy the serial port as is the case with PC/SC.

At this stage we have to be very careful with the above observations as they are really based on the aforementioned specific implementations. In order to obtain a clearer picture on what are the actual issues involved around the performance of each technology, it is recommended that the reader carefully examines the timings in each table and for each individual function. In that way any potential influence to the overall result by each individual function is removed.

4.2 Further Discussion of the Results

When comparing the performance of the Sm@rtCafé implementation under PC/SC in the two different readers we realise that the GCR implementation appears to perform on average better.

When comparing the GemXpresso implementations in the two available readers and under PC/SC we observe that on average the GCR implementation is faster than the PCT. On the other hand the corresponding standard deviation of the GemXpresso and Sm@rtCafé implementation on the PCT reader is lower. For both smart card implementations under PC/SC it appears that application selection takes place faster in the GCR reader.

It is worth mentioning that both the “Connect” and “Disconnect” figures are extremely small, this can be possibly explained by the fact that in PC/SC there is constant traffic in the serial port. Therefore, connecting and disconnecting to/from that card happens almost immediately. Overall, the GemXpresso implementation under PC/SC on the GCR reader maintains a marginal lead. The fact that both smart card implementations demonstrate slower measurements in the PCT reader can be explained, at this stage, due to inefficient APDU handling either at the corresponding card reader or at the PC/SC driver level or even due to differences in the actual smart card microprocessors.

When comparing the performance of the Sm@rtCafé implementation under OCF we realise that the GCR implementation is significantly faster. When comparing the performance of GemXpresso implementation under OCF we observe that the GCR implementation is once more relatively faster compared with the PCT implementation

Both smart card implementations under OCF on the PCT reader demonstrate notably large figures for the “Connect and” “Select” functions along with unreasonably large standard deviations when compared with the corresponding ones on the GCR reader. This indicates that probably the OCF driver for the PCT reader is not properly implemented.

5. CONCLUSIONS AND DIRECTIONS FOR FURTHER RESEARCH

When checking the typical smart card cryptographic algorithm figures from Table 1 we can see that, a typical cryptographic operation ranges from 20-450ms and on average it takes around 160ms. This figure is equivalent with sending 10 bytes to the card or sending 10 bytes and also getting 10 bytes as a response. Analogous conclusions can be drawn when taking into account the fact that the performance of different smart card cryptographic algorithms is comparable with sending or receiving a number of bytes to/from the smart card.

Up to recently, a lot of the discussion about smart cards concentrated on improving the performance of the smart card cryptographic functions. The end-result was tiny improvements in order of a couple of tens of milliseconds for a cryptographic function that could be used once or twice within a smart card application. It is clear that more effort should be placed on improving the smart card communication API, as it appears to be more extensively used during the execution of a smart card application, than just concentrating on improving the performance of the smart card cryptographic algorithms.

We have to bear in mind that the PC/SC terminal application was developed in Java. If it was developed in C++ or Visual Basic, non-interpreted languages, then it could be the case that the overall execution time is improved. On the other hand the portability issue will be eliminated, as the terminal application will be closely tied in with the underlying development platform.

Further work is actually required in order to obtain more results with the latest versions of the smart card reader drivers and APIs (particularly the new version of OCF 1.2). It would also be helpful to obtain more results when testing the proposed functionality with the native smart card readers in order to explore the actual benefits from sacrificing speed against interoperability.

Another important factor which significantly reduces the overall smart card application performance, and has not yet received the necessary attention, is the size of the communication buffer, i.e. the APDU buffer. More effort should be placed in order to increase the size of the buffer especially in the light of the multi applications smart cards and the high probability of large packets of information travelling towards the card, e.g. applications to be downloaded. For example with an APDU buffer of 512 bytes an application will be downloaded in significantly less time and with less APDU exchanges compared to a 256 byte buffer.

A final remark is that it is not easy to talk about absolute timings and performance measurements when smart card communication is involved. Improving the performance of smart card cryptographic functions [24,25] used to be the area that received the most attention. As demonstrated by this paper increasing emphasis should also be placed in additional areas.

ACKNOWLEDGEMENTS

The author would like to thank Dieter Gollmann for his helpful comments. Also, Konstantinos Skourtis for his help regarding the development of the PC/SC terminal applications. Finally, his current employer, Visa International EU, for facilitating the authors' participation in the conference although the views of the paper do not necessarily represent the company's view.

REFERENCES

- [1] Gemplus, "Multi_Application Chip Operating System (MCOS) Reference Manual Ver 2.2", 1990.
- [2] Gemplus, "Multi_Application Payment Chip Operating System (MPCOS) Reference Manual Ver 2.2", 1994.
- [3] General Information Systems Ltd., "OSCAR, Specification of a smart card filling system incorporating data security and message authentication", Available from <http://www.gis.co.uk/oscmn1.htm>
- [4] Schlumberger, "Cyberflex Smart card Series Developers manual", Available from <http://www.cyberflex.austin.et.slb.com/cyberflex/cyberhome>
- [5] Gemplus, "GemXpresso Reference Manual", Available from <http://www.gemplus.fr/gemxpresso/index.htm>
- [6] Constantinos Markantonakis, "The Case for a Secure Multi-Application Smart Card Operating System", Information Security Workshop 97 (ISW'97), September 17-19, 1997, Ishikawa, Japan, In Lecture Notes in Computer Science (LNCS), volume 1396, pp.188-197
- [7] MAOSCO, "MULTOS Reference Manual Ver 1.2", Available from <http://www.multos.com/>
- [8] PC/SC Workgroup, "Specifications for PC-ICC Interoperability", Available from www.smartcardsys.com
- [9] Open Card Consortium, "OpenCard Framework Specification OCF", Available from www.opencard.org
- [10] International Standard Organisation. ISO/IEC 7816-2, "Identification cards --- Integrated circuit(s) cards with contacts, Part 2, Dimensions and location of the contacts", International Organization for Standardisation, 1996.
- [11] International Standard Organisation. ISO/IEC 7816-3, "Identification cards --- Integrated circuit(s) cards with contacts, Part 3, Electronic signals and transmission protocols", International Organization for Standardisation, 1997.
- [12] W. Rankl, W. Effing, "Smart Card Handbook", John Wiley and Sons, 1997.
- [13] Sun Microsystems, Java Card 2.0 Language Subset and Virtual Machine Specification, <http://www.javasoft.com/products/javacard>, 1998.
- [14] Sun Microsystems, Java Card 2.0 Programming Concepts, <http://www.javasoft.com/products/javacard>, 1998.
- [15] Sun Microsystems, Java Card 2.0 The Java Card API Ver 2.1 Specification, <http://www.javasoft.com/products/javacard>, 1998.
- [16] Gemplus, GemXpresso Reference Manual, Gemplus, 1998.
- [17] Giesecke & Devrient, Sm@rtCafe Reference Manual, Sm@rtCafé Professional Toolkit, 1999.
- [18] International Standard Organisation. ISO/IEC 7816-4, "Information technology --- Identification cards --- Integrated circuit(s) cards with contacts --- Interindustry Commands for Interchange", International Organization for Standardisation, 1995.
- [19] Gemplus Web Site, www.gemplus.com/developers, 1999.
- [20] Towitoko Web Site, http://www.towitoko.de/deutsch/eng/WD1034_s.htm, 1999.
- [21] Mary Kirtland, "The COM Programming Model Makes it Easy to Write Components in Any Language", Technical Report, Microsoft Systems Journal, December 1997, <http://www.microsoft.com/msj/1297/complus2/complus2.htm>.
- [22] Mark Russinovich, "Portmon", <http://www.sysinternals.com>, 1999.

- [23] Helena Handschuch, Pascal Paillier, "Smart Card Cryptoprocessors for Public Key Cryptography", In Springer Verlag. Third Smart Card Research and Advanced Application Conference - CARDIS'98, September 1998 to be published.
- [24] R. Ferreira, R. Malzahn, P. Marissen, J.-J. Quisquater and T. Wille: "*FAME: A 3rd generation coprocessor for optimising public key cryptosystems in smart card applications*" In P. H. Hartel et al., eds, *Smart card Research and Advanced Applications -- Cardis '96*, Amsterdam (The Netherlands), 18-20th September 1996, Publ. Stichting Mathematisch Centrum, pp. 59-72, 1996.
- [25] T. Boogaerts, "*Implementation of Elliptic curves cryptosystems for smart cards*", In proc. of *CARDIS 1998*, 14-16th September 1998.
- [26] Constantinos Markantonakis, "Interfacing with Smart card Applications (The PC/SC and Open Card Framework)", Elsevier Information Security Technical Report, 3(2):82-89, 1999.

This Page Intentionally Left Blank

MODELLING AUDIT SECURITY FOR SMART-CARD PAYMENT SCHEMES WITH UML-SEC

Jan Jürjens*

Computing Laboratory

University of Oxford

GB

<http://www.jurjens.de/jan>

jan@comlab.ox.ac.uk

Abstract To overcome the difficulties of correct secure systems design, we propose formal modelling using the object-oriented modelling language UML. Specifically, we consider the problem of accountability through auditing.

We explain our method at the example of a part of the Common Electronic Purse Specifications (CEPS), a candidate for an international electronic purse standard, indicate possible vulnerabilities and present concrete security advice on that system.

1. INTRODUCTION

Designing secure systems correctly is difficult. Many flaws have been found in proposed security-critical systems and protocols, sometimes years after their publication (e.g. [Low96]). This motivates using formal concepts and tools developed for systems design to ensure fulfillment of security requirements.

In this work we concentrate on *accountability* and the enforcement of *audit policy*, which provides the requirements for record keeping.

The Unified Modeling Language (UML) [RJB99] is an industry standard language for specifying software systems. Following [Jür01c], we use a simplified formal core of UML (for which [Jü01c] gave an extension with security

*Supported by the Studienstiftung des deutschen Volkes and the Computing Laboratory.

primitives called UMLsec) extended to model and investigate a security-critical part of the Common Electronic Purse Specifications (CEPS) [CEP00]. CEPS is a candidate for a globally interoperable electronic purse standard and is supported by organisations (including Visa International) representing 90 percent of the world's electronic purse cards, making its security an important goal.

A more general aim of this line of research started in [Jür01c, Jür01d] is to use UML to encapsulate knowledge on prudent security engineering and thereby make it available to developers not specialized in security.

In the following subsection we present some background information and refer to related work. In Section 3, we give an overview over the Common Electronic Purse Specifications, specify the part under consideration, explain the security threat model and give results. We end with a conclusion and indicate further planned work.

1.1. SECURITY-ASSURANCE USING FORMAL MODELLING

There has been extensive research in using formal models to verify secure systems. A few examples are [BAN89, Low96, Pau98, Jür00, AJ01, Jür01a, WW01], for an overview wrt. security protocols cf. [GSG99, RSG⁺01]. However, auditing does not seem to have been considered extensively.

An overview on payment systems is given in [AJSW00]. Smart card protocols have been investigated using formal logic in [ABKL93]. [BCG⁺00] considers secure information flow between applets in a multi-application smart-card. A different part of the CEPS is investigated in [JW01] using the CASE-tool AUTO FOCUS.

While many case-studies consider security protocols from the academic literature (usually presented in a much more tractable form), a notable example of a verification of a smart-card payment system used in practice can be found in [And99]. Also, [SCW00] gives a detailed, formal proof of a Smartcard product for electronic commerce.

Object-oriented systems offer a very suitable framework for considering security due to their encapsulation and modularisation principles [Eck95, Bd-VFS98, Sam00]. In [OvS94] the authors formulate a taxonomy for security in object-oriented databases. An object-oriented data flow model for smart card security is given in [GHdJF96].

2. MODELLING OBJECT-ORIENTED SECURITY

We use a simplified fragment of the visual modeling language UML (the industry-standard in object-oriented modelling), following [Jür01c].

UML consists of several diagram types describing different views on a system. Here we concentrate on using the UML notation to specify security requirements on auditing mechanisms of a system.

We use the following two kinds of diagrams:

Class diagrams define the static structure of the system: classes with attributes and operations/signals and relationships between classes. We use them to specify how the objects may communicate.

Statechart diagrams give the dynamic behaviour of an individual object: input events may cause state in change or (output) actions.

Below we will define the (simplified) abstract syntax for these two kinds of diagrams (on which the formal reasoning relies). Later we will also use the usual diagrammatic notation for readability.

We define the data type **Exp** of cryptographic messages that can be exchanged between objects. We assume a set D of basic data values. The set **Exp** contains the expressions defined inductively by the grammar

$E ::=$	expression
d	data value ($d \in D$)
K	key ($K \in \mathbf{Keys}$)
x	variable ($x \in \mathbf{Var}$)
(E_1, \dots, E_n)	concatenation
$\text{Enc}(K, E)$	encryption ($K \in \mathbf{Keys} \cup \mathbf{Var}$)
$\text{Dec}(K, E)$	decryption ($K \in \mathbf{Keys} \cup \mathbf{Var}$)
$\text{Mac}(K, E)$	MAC ($K \in \mathbf{Keys} \cup \mathbf{Var}$)
$\text{Ver}(K, E)$	verify MAC ($K \in \mathbf{Keys} \cup \mathbf{Var}$)

The part of the CEPS considered here uses symmetric encryption. As usual, we assume the equations $\text{Dec}(K, \text{Enc}(K, E)) = E$ and $\text{Ver}(K, \text{Mac}(K, E)) = E$ and assume that no equations except those following from these hold.

2.1. CLASS DIAGRAMS

We first give the definition for class models.

An *attribute specification* $A = (\text{att_name}, \text{att_type})$ is given by a name att_name and a type att_type .

An *operation specification* $O = (\text{op_name}, \text{Arguments op_type})$ is given by a name op_name , a set of Arguments , and the type op_type of the return value. Note that the set of arguments may be empty, and that the return type may be the empty type \emptyset denoting absence of a return value. An *argument* $A = (\text{arg_name}, \text{arg_type})$ is given by its name arg_name and its type arg_type .

A *signal specification* is just like an operation specification, except that there is no return type.

An *interface* $I = (\text{int_name}, \text{Operations}, \text{Signals})$ is given by a name `int_name` and sets of operation names `Operations` and signal names `Signals` specifying the operations and signals that can be called resp. sent through it.

A *class model* $C = (\text{class_name}, \text{Stereotypes}, \text{AttSpecs}, \text{OpSpecs}, \text{SigSpecs}, \text{Interfaces})$ is given by a name `class_name`, a set of `Stereotypes` (for our present purposes, this may be empty or contain the stereotype «*log*»), a set of attribute specifications `AttSpecs`, a set of operation specifications `OpSpecs`, a set of signal specifications `SigSpecs` and a set of class interfaces `Interfaces`.

A *class diagram* $D = (\text{Cls}, \text{Dependencies})$ is then given by a set `Cls` of class models and a set of `Dependencies`. A *dependency* is a tuple (`client`, `supplier`, `interface`, `stereotype`) consisting of class names `client` and `supplier` (signifying that `client` depends on `supplier`), an interface name `interface` (giving the interface of the class `supplier` through which `client` accesses `supplier`; if the access is direct this field contains the `client` name) and a `stereotype` which for our present purposes will be «*send*». We require that the names of the class models are mutually distinct.

In the diagrammatic notation (cf. Figure 1), a class model is represented by a rectangle with three compartments giving its name, its attributes and its operations (since all values are of type **Exp**, the type information is omitted in the diagrams given in this paper for readability).

The concurrently executed objects communicate asynchronously by exchanging signals, possibly with arguments. Dependency arrows marked with «*send*» from a class C to a class C' indicate that (an object instance of) C may send a signal to (an object of) C' . If the arrow points to an interface of C' (represented by a circle attached to the class rectangle), C may only use the signal listed in the corresponding interface specification (the respective rectangle marked «*inter face*»). For example, in Figure 1 `Card` may send the signal `CLog` with arguments `dt`, `lda`, `m`, `nt`, `bal`, `s2` to `CardLog`, and `Issuer` may send `Respl` with arguments `ceps`, `iss`, `lda`, `s2` to `LSAM` (but not the other signals offered by the `LSAM`, since they are reserved for `Card`).

2.2. STATECHART DIAGRAMS

We fix a set `Var` of (typed) variables x, y, z, \dots used in statechart diagrams.

We define the notion of a statechart diagram for a given class model C : A *statechart diagram* $S = (\text{States}, \text{init_state}, \text{Transitions})$ is given by a set of `States` (that includes the initial state `init_state`) and a set of `Transitions`.

A *statechart transition* $t = (\text{source}, \text{event}, \text{guard}, \text{Actions}, \text{target})$ is given by a `source` state, an operation term `op_term`, a `guard`, a list of `Actions` and a `target` state. Here an *event* is the name of an operation or signal with a list of distinct variables as arguments that is assumed to be well-typed (e.g. `op(x, y, z)`). Let the set `Assignments` consist of all partial functions that assign

to each variable and each attribute of the class C a value of its type (partiality arises from the fact that variables may be undefined). A *guard* is a function $g : \text{Assignments} \rightarrow \text{Bool}$ evaluating each assignment to a boolean value. An *action* can be either to assign a value v to an attribute a (written $a := v$), to call an operation op resp. to send a signal sig with values v_1, \dots, v_n (written $\text{op}(v_1, \dots, v_n)$ resp. $\text{sig}(v_1, \dots, v_n)$), or to return values v_1, \dots, v_n as a response to an earlier call of the operation op (written $\text{return}_{\text{op}}(v_1, \dots, v_n)$). In each case, the values can be constants, variables or attributes (and need to be well-typed). In the case of *output actions* (calling an operation or sending a signal) we include the types of the arguments (and possibly of the return value).

To formally reason about statecharts, [Jür01c] gives a formal behavioural semantics (which has to be omitted here).

In the diagrammatic notation (cf. e.g. Figure 2), the states in a statechart are represented by rectangles, where the initial state has an ingoing transition from the start marker (a full circle). As specified in the abstract syntax, the transitions between states can carry three kinds of information as labels:

Events are names of operations provided by the class together with argument variables (e.g. $\text{RespI}(ic, cep, ex, nt, s1)$ in Figure 2). If another object sends a signal, the corresponding transition is triggered, and the variables are bound to the arguments given. If a variable has already been assigned a value at an earlier point in the execution of the state machine, the transition is only executed if the two values match (i. e. an implicit equality conditional is enforced).

Guards are conditionals written in square brackets (e.g.

$[\text{Ver}(K_l, s2) = (bal, cep, , iss, nt, s1) \wedge \dots]$ in Figure 3). A transition can only be triggered if all labeling guards are fulfilled. Sometimes a guard involves a variable that has not been assigned a value before (e.g. as an argument of an input event). Since in our behavioural formal semantics we implicitly quantify over free variables, this means that the equation *assigns* the corresponding value to the free variable and to make this clear we write the equation then as “:=” (but formally there is no difference to the usual “=”). An example is $[ml := \text{Mac}(r, (ic, cep, nt, lda, m, s1))]$ in Figure 2. Note that this is different from an action that assigns a value to an attribute; the variables here are local to the statechart diagram and are merely syntactic means for describing the object behaviour.

Actions are names of operations provided by other classes, written with a preceding backslash and including arguments (e.g. $\backslash\text{Init}(dt, lda, m)$ in Figure 2). If a transition is fired, all labeling actions are executed, which means that the objects supplying the operations are called with the respective arguments.

E.g., in Figure 4, the transition from `Init` to `Load` is fired when the signal `Load` is sent and certain validity conditions are fulfilled. Then in turn the signal `Respl` is sent.

2.3. MODELLING SYSTEMS

We model a system S by a class diagram D and a set of statechart diagrams S , one for each object. In general, we also use *deployment diagrams* e.g. to distinguish secure from insecure communication links [Jür01c]. We omit these here because all links between the participants in the CEPS load transaction considered below are insecure.

We briefly sketch how to formally interpret such system models (for more details cf. [Jür01c]). When interpreting a system model S , each operation, say `op`, communicating along an insecure dependency is replaced by an operation `op_out` (for actions) resp. `op_in` (for events). An adversary A is a state machine with actions `op_in` and events `op_out` (for each operation `op` in S communicating insecurely). We only consider adversaries that are computationally bounded in the sense that they can encrypt or decrypt messages only when in possession of the relevant key (for a formalisation of this concept cf. [Jür01b]).

Output values are buffered without preserving the order of messages (i. e. buffers are multi-sets). Values without specified transition in an object are ignored. In both these assumptions we follow the usual UML point of view.

Histories are sequences of states of all state machines corresponding to the objects, and buffer contents (where the state machines for the specified objects are derived from the statechart diagrams as defined in [Jür01c]).

Given a system model S and an adversary A , the *execution of S in presence of A* is given as the set of *possible histories*.

A history \vec{h} is a possible history if

- in its first component \vec{h}_0 , all states are initial states and the buffer is empty, and if
- for each $n \geq 0$ and each class model $C \in \mathbf{Cls} \cup \{A\}$ that changes state at time n , there is a transition $t_{C,n}$ from its state at n to its state at $n + 1$ such that for given n the multiset of (input) events ϵ_n corresponding to the transitions $(t_{C,n} : C \in \mathbf{Cls})$ is contained in the buffer content B_n at n and $B_{n+1} = (B_n \setminus \epsilon_n) \cup A_n$ (for the multiset A_n of (output) actions fired by the transitions $\{t_{C,n} : C \in \mathbf{Cls}\}$).

2.4. AUDITING

We incorporate auditing in our framework by specifying a subset $\mathbf{Audit} \subseteq \mathbf{Cls}$ of class models used to store the audit data.

For completeness we give the following general definition of secure auditing. Note that the definition only applies to the situation where all the objects in the system model are honest. Thus in the considerations on CEPS below we need more specific notions of secure auditing.

Definition 1 A system model S provides *secure auditing* if, in presence of any adversary, the corresponding attribute values of all audit objects coincide when all objects have reached a final state.

Note that here we do not consider the question whether an object may be kept from reaching its final state.

3. CEPS

We give an overview over the Common Electronic Purse Specifications.

Stored value smart cards (“electronic purses”) have been proposed to allow cash-free point-of-sale (POS) transactions offering more fraud protection than credit cards: Their built-in chip can perform cryptographic operations which allows transaction-bound authentication (while credit card numbers are valid until the card is stopped, enabling misuse). The card contains an account balance that is adjusted when loading the card or purchasing goods.

The Common Electronic Purse Specifications (CEPS) define requirements for a globally interoperable electronic purse scheme providing accountability and auditability. The specifications outline overall system security, certification and migration. For more detail on the functionality of CEPS cf. [CEP00].

Here we consider a central part of CEPS, the (unlinked, cash-based) load transaction, which allows the cardholder to load electronic value onto a card in exchange for cash at a load device belonging to the load acquirer. The participants involved in the transaction protocol are the customer’s card, the load device and the card issuer. The load device contains a Load Security Application Module (LSAM) that is used to store and process data (and is assumed to be tamper-resistant). During the transaction, the account balance in the card is incremented, and the amount is logged in the LSAM and sent to the issuer for later financial settlement between the load acquirer and the card issuer.

3.1. SPECIFICATION OF CEPS LOAD TRANSACTION

We give a specification of the CEPS load transaction (slightly simplified by leaving out security-irrelevant details, and also leaving out details needed for exception processing and declined loads). Load transactions in CEPS are on-line transactions using symmetric cryptography for authentication. We only consider unlinked load (where the cardholder pays cash into a (possibly unat-

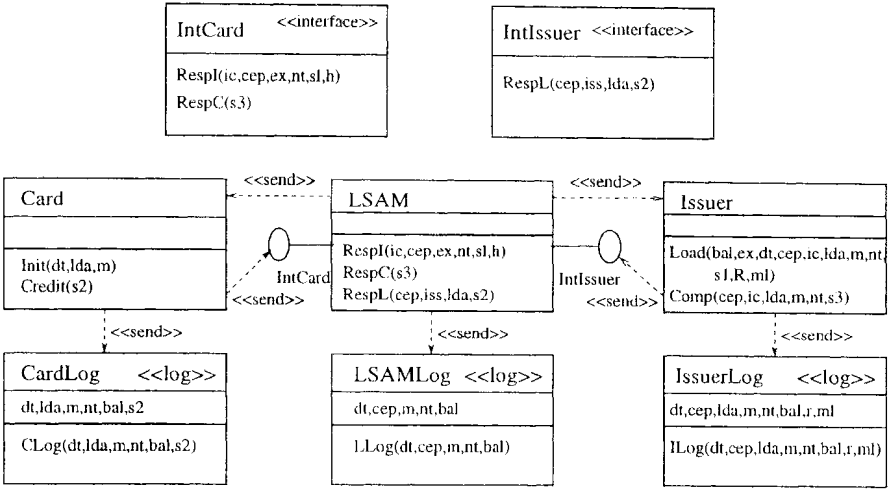


Figure 1 Class diagram for Load transaction

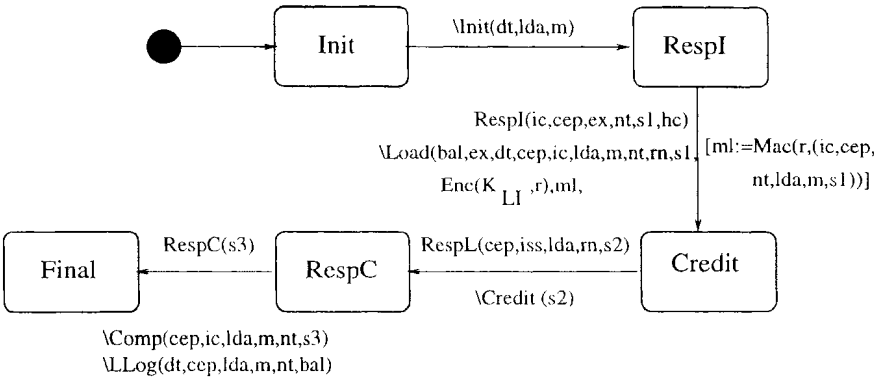


Figure 2 Statechart for LSAM

tended) loading machine and receives a corresponding credit on the card) since linked load (where funds are transferred e.g. from a bank account) offer fewer possibilities for fraud [CEP00, Funct. Req. p. 12]. We use class diagram and statechart diagrams introduced above.

First, we give the involved classes and their dependencies in the class diagram in Figure 1. For the participants of the protocol, we have the classes Card, LSAM, and Issuer. Also, each of the three classes has an associated class used for logging transaction data (marked with the stereotype `<<log >>`).

We specify the behaviour of the classes Card, LSAM, and Issuer using UML statecharts in the remaining figures.

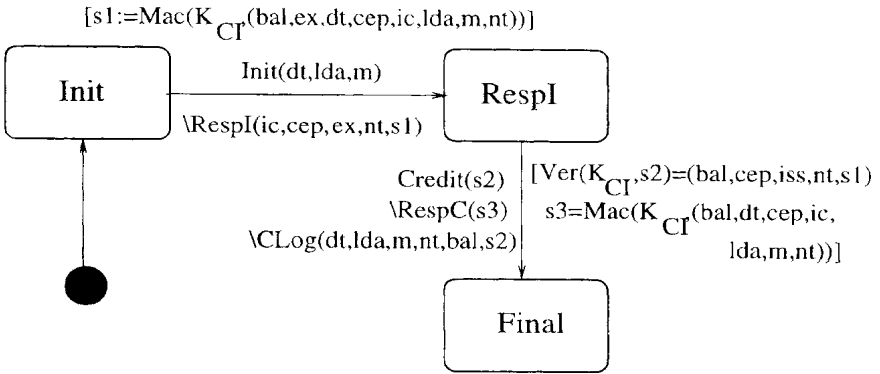


Figure 3 Statechart for card

The LSAM (Figure 2) initiates the transaction after the CEP card is inserted into the load device, by sending the “Init for load” message `Init` with arguments the transaction date and time dt , the load device identifier lda and the transaction amount m (which is the amount of cash paid into the load device by the card holder that is supposed to be loaded onto the card). Whenever the card (Figure 3) receives this message after being inserted into the load device, it sends back the “Init for load response” message `RespI` to the LSAM, with arguments the card issuer identifier ic (as stored on the card), the card identifier cep , the balance (prior to load) bal , the card expiration date ex , the card’s transaction number nt unique to the transaction, and the card MAC $s1$. $s1$ consists of the values ex , bal , dt , cep , ic , lda , m and nt , all of which are signed with the key K_C^{-1} shared between a particular card and the corresponding card issuer. The LSAM then sends to the issuer the “load request” message `Load` with arguments bal , ex , dt , cep , ic , lda , m , nt , rn , $s1$, $\text{Enc}(K_{LI}, r)$, and ml . rn is the reference number assigned by the LSAM to the transaction. $\text{Enc}(K_{LI}, r)$ is the encryption of a random number r generated by LSAM under a key K_{LI} shared between the LSAM and the issuer. ml is the MAC of the following data using the fresh key r generated by the LSAM: ic , cep , nt , lda , m , and $s1$. The issuer (Figure 4) checks if ic is a valid issuer identifier, cep a valid card identifier and the expiration date ex has not been exceeded. The issuer verifies if $s1$ is a valid MAC generated from the values ex , bal , dt , cep , ic , lda , m and nt with the key K_{CI} (i. e. if $\text{Ver}(K_{CI}, s1) = (bal, ex, dt, cep, ic, lda, m, nt)$). The issuer retrieves r from $\text{Enc}(K_{LI}, r)$ (using the key K_{LI} shared between the LSAM and the issuer, i.e. $r := \text{Dec}(K_{LI}, R)$) and checks if ml is a valid MAC of the values ic , cep , nt , lda , m , and $s1$ using the key r , i. e. if $\text{Ver}(ml, r) = (ic, cep, nt, lda, m, s1, hc)$. Lastly, the issuer checks that the key K_{LI} is actually shared with the LSAM named lda (we write this as $\text{Shared}(K_{LI}) = lda$ assuming a function `Shared`

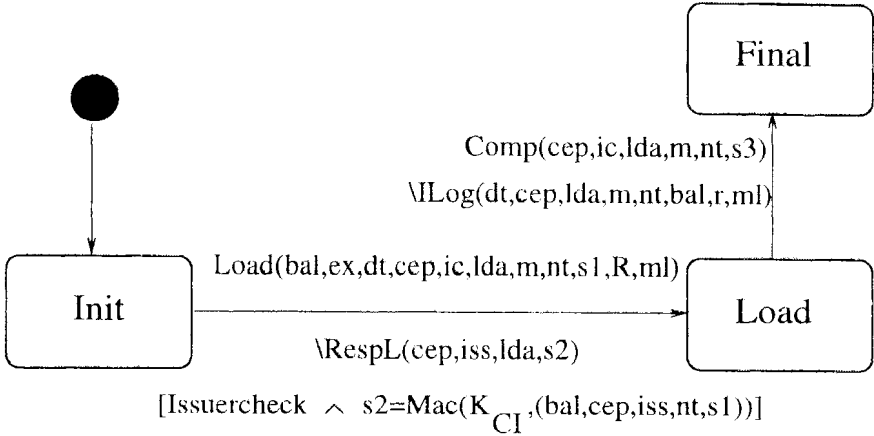


Figure 4 Statechart for Issuer

which assigns LSAMs to keys). If all these checks succeed (which in Figure 4 are abbreviated by the conditional *Issuercheck*), the issuer sends the “respond to load” message *RespL* with arguments *cep*, *ic*, *lda*, *m*, and *s2* to the LSAM. *s2* consists of the following values, signed with the key K_{CI} : *bal*, *cep*, *iss*, *nt*, and *s1*.

Next, the LSAM sends the “credit for load” message *Credit* with argument *s2* to the card. Finally, the card (on successful verification of *s2*) answers by sending the “response to credit for load” message *RespC* with argument *s3* back to the LSAM. *s3* consists of the following values, signed with the key K_{CI} : *bal*, *dt*, *cep*, *ic*, *nt*, *lda*, *m*, and *nt*. The card also sends the logging message *CLog* to the object *CardLog*, with arguments *dt*, *lda*, *m*, *nt*, *bal*, and *s2*. Finally, the LSAM sends to the issuer the “transaction completion message” *Comp* with arguments *cep*, *ic*, *lda*, *m*, and *nt*. Also, the LSAM sends the logging message *LLog* to the object *LSAMLog*, with arguments *dt*, *cep*, *iss*, *m*, *nt*, and *bal*. On receipt of the message *Comp* from the LSAM (and provided the contained values match the corresponding values communicated earlier), the issuer sends the logging message *ILog* to the object *IssuerLog*, with arguments *dt*, *cep*, *lda*, *m*, *nt*, and *bal*.

The logging objects simply take the arguments of their operations and update their attributes accordingly.

3.2. SECURITY THREAT MODEL

We consider the threat scenario for the load transaction and derive audit security conditions. The general assumption is that the card, the LSAM and the security module of the card issuer are tamper-resistant (in particular that the

contained secret keys cannot be retrieved). The protocol can be attacked e.g. by inserting adapters or relays between the LSAM and the card loading device or by intercepting the communication with the card issuer.

We concentrate on the load acquirer as a possible attacker of the transaction. The cardholder could try to attack the protocol by interrupting it e.g. by pulling out the card (thus one needs to make sure that money is not returned to the cardholder after the card has been loaded) or could try to duplicate the loaded money by loading it on two cards simultaneously using an adapter (at an unattended load device). We do not consider these kinds of attacks here. Also, the card issuer is not so interesting as an attacker since she controls the settlement scheme that is performed after the transactions, so the cardholder and the load acquirer have to trust her to some degree anyway (and my disputes would have to be settled in court).

Given the participants of the protocol, the load acquirer can attack either the cardholder, or another load acquirer, or the card issuer, with the goal either to keep the amount paid by the cardholder (and not have to pass it on to the card issuer), or to credit a card owned by the load acquirer himself without having to pay any money to the card issuer.

We consider attacks against the cardholder. Smart cards can not communicate directly with the cardholder. Thus there is the usual threat that a load device (possibly belonging to a corrupt load acquirer) is manipulated so that the transaction is performed as if the cardholder had only paid part of the amount that was actually paid, or so that the transaction is not performed at all. Then the load acquirer would not have to pay the amount to the card issuer. However, we assume that the cardholder can verify after the transaction if the correct amount has been loaded (possibly using a portable card reader), and that a complaint settlement scheme settles any disputes arising from such attacks. The correct functioning of the settlement scheme relies on the fact that the cardholder should only be lead to believe (e.g. when checking the card with a portable card reader) that a certain amount has been correctly loaded if he is later able to *prove* this using the card – otherwise the load acquirer could first credit the card with the correct amount, but later in the settlement process claim that the cardholder tried to fake the transaction. Thus we have to check the following audit security condition on the attributes of **CardLog** after **Card** has reached its final state:

Correct amount: s_2 and s_1 verify correctly (say $\text{Ver}(K_{CI}, \text{CardLog}.s_2) = (bal', cep', iss', nt', s_1')$ and $\text{Ver}(K_{CI}, s_1') = (bal'', ex'', dt'', cep'', ic'', lda'', m'', nt'')$ for some values bal', bal'', \dots), and additionally we have $\text{CardLog}.m = m''$ (i. e. the correct amount is logged).

A load acquirer could also try to attack the protocol in order to masquerade as another load acquirer for the purpose of the settlement process, in order not

to pay the amount paid in by the cardholder to the card issuer. To prevent this, we need to ensure the following audit security condition:

No masquerade: We have $\text{Shared}(K_{LI}, \text{IssuerLog}.lda)$.

ml is supposed to provide a guarantee that the load acquirer owes the transaction amount to the card issuer [CEP00, Funct.spec., 6.6.1.6]. To be able to make use of this guarantee, the card issuer needs to be able to show that her possession of the guarantee implies that the load acquirer owes her the amount (and that the card issuer could not just produce ml himself). Thus we have the audit *functionality* condition

Acquirer guarantee functionality: If

$$\text{IssuerLog}.ml = \text{Mac}(\text{IssuerLog}.r, (ic', cep', nt', lda', m', s1', hl'))$$

then the LSAM lda' has received m' .

Also, we would like to ensure that this guarantee is always given, i. e. that the following audit security condition (the converse of the above functionality condition) is fulfilled:

Acquirer guarantee security: If the state machines of card and card issuer have reached the final state and $\text{CardLog}.m = m'$ then

$$\text{IssuerLog}.ml = \text{Mac}(\text{IssuerLog}.r, (ic', cep', nt', lda', m', s1', hl'))$$

Note that the precondition that card and card issuer have reached their final states is necessary. In particular, if the load device simply takes the inserted cash without taking any further action, the cardholder has no proof of this (but this is the usual risk taken at automatic purchase machines), and if the LDA does not complete its last action, exception processing on the side of the card issuer would have to be followed (not considered here).

3.3. RESULTS

Theorem 1 Acquirer guarantee functionality is not provided in the proposed scheme.

The reason for this is that the security of the data elements in ml are only protected by the random value r , which in turn is communicated encrypted under the secret key K_{LI} shared between load acquirer and card issuer. This means that the card issuer would in principle be capable of manufacturing ml and r herself. Therefore possession of ml does not suffice for the issuer to be able to prove that the load acquirer manufactured ml .

This is not a serious threat since one would expect that in practical situations any dispute arising from this could be resolved in a settlement process. However,

the CEPS explicitly postulate this requirement. This should either be clarified, or the data element *ml* be changed to involve a signature with a private key of the load acquirer.

Theorem 2 *The audit security conditions **Correct amount, No Masquerade, and Acquirer guarantee security** are fulfilled.*

The formal proof of this theorem has to be omitted for space limitations and will be included in the long version of the paper. The proof proceeds inductively along the lines of ideas in [Pau98] and uses results in [Jür01b, Jür01a]. Here we can only give some informal remarks:

Correct amount: Essentially, one has to show that the key K_{CI} shared between the card and the card issuer established end-to-end security between card and issuer.

No masquerade: This amounts to showing that the load device identifier, as stored in the issuer log, corresponds to the load device with which the issuer shares the key K_{LI} .

Acquirer guarantee security: Here one has to show that the integrity of the information passed between card and card issuer is preserved.

4. CONCLUSION AND FUTURE WORK

We investigated the security of the currently developed Common Electronic Purse Specifications (CEPS) using the object-oriented modelling language UML. Benefits of our approach include the possibility to investigate security in the context of general system development. Since security violations often occur at the boundaries between security mechanisms (such as protocols) and the general system [And94], this is very helpful. We choose UML among the various object-oriented modelling languages since it is the current de-facto industry standard and thus many developers will be able to take advantage of an extension of UML by security primitives.

Apart from these methodological benefits, this work delivers concrete results on the security of the payment systems that are to be developed and fielded according to the CEPS. Our investigation exhibited a weakness arising from the fact that the card issuer does not obtain a sound proof of transaction from the load acquirer. As usual, the positive results given here should not be interpreted as proving the CEPS secure (as well-known, such a proof is impossible).

Due to space constraints we could only consider one part of the CEP specifications, the other parts are left for further work. Since UML offers a variety of modelling mechanisms with varying degrees of abstraction, considering a large part of a system seems relatively feasible. It may also be interesting to

consider reevaluation of security after system changes. Also, we will extend this approach beyond reasoning about accountability.

Acknowledgments

This idea to use UML to specify security properties arose when doing security consulting for a project during a research visit with M. Abadi at Bell Labs (Lucent Tech.), Palo Alto, whose hospitality is gratefully acknowledged. Comments or advice from participants of the summer school “Foundations of Security Analysis and Design 2000” and the Dagstuhl seminar “Security through Analysis and Verification” (especially D. Gollmann) are gratefully acknowledged, as well as useful comments from G. Wimmel and the anonymous referees on a draft.

References

- [ABKL93] M. Abadi, M. Burrows, C. Kaufman, and B. Lampson. Authentication and delegation with smart-cards. *Science of Computer Programming*, 21(2):93 – 113, 1993.
- [AJ01] M. Abadi and Jan Jürjens. Formal eavesdropping and its computational interpretation, 2001. Submitted.
- [AJSW00] N. Asokan, P. Janson, M. Steiner, and M. Waidner. The state of the art in electronic payment systems. *Advances in Computers*, 53, 2000.
- [And94] R. Anderson. Why cryptosystems fail. *Communications of the ACM*, 37(11):32–40, November 1994.
- [And99] R. Anderson. The formal verification of a payment system. In Mike Hinchey and Jonathan Bowen, editors, *Industrial-Strength Formal Methods in Practice*, pages 43–52. Springer, 1999.
- [BAN89] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *Proc. Royal Society of London A*, 426:233–271, 1989.
- [BCG⁺00] P. Bieber, J. Cazin, P. Girard, J.-L. Lanet, V Wiels, and G. Zanon. Checking secure interactions of smart card applets. In *ESORICS*, 2000.
- [BdVFS98] E. Bertino, S. De Capitani di Vimercati, E. Ferrari, and P. Samarati. Exception-based information flow control in object-oriented systems. *ACM Transactions on Information and System Security*, 1 (1): 26–65, 1998.
- [CEP00] CEPSCO. Common Electronic Purse Specifications, 2000. Business Requirements vers. 7.0, Functional Requirements vers. 6.3, Technical Specification vers. 2.2, available from <http://www.cepsco.com>.
- [CFMS94] S. Castano, M. Fugini, G. Martella, and P. Samarati. *Database Security*. Addison Wesley, 1994.
- [Eck95] C. Eckert. Matching security policies to application needs. In J. H.P. Eloff and S.H. von Solms, editors, *IFIP TC11 11th International Conference on Information Security*, pages 237–254. Chapman & Hall, 1995.
- [GHdJF96] H. Glaser, P. Hartel, and E. de Jong Frz. Structuring and visualising an IC - card security standard. In *in [HPQ96]*, pages 89–110, 1996.
- [GSG99] Stefanos Gritzalis, Diomidis Spinellis, and Panagiotis Georgiadis. Security protocols over open networks and distributed systems: Formal methods for their analysis, design, and verification, *Computer Communications Journal*, 22(8):695–707, 1999.
- [HPQ96] P. H. Hartel, P. Paradinas, and J. - J. Quisquater, editors. *2nd Smart card research and advanced application conference (CARDIS)*. Stichting Mathematisch Centrum, Amsterdam, 1996.

- [Jür00] Jan Jürjens. Secure information flow for concurrent processes. In *CONCUR 2000 (11th International Conference on Concurrency Theory)*, volume 1847 of *LNCS*, pages 395–409, Pennsylvania, 2000. Springer.
- [Jür01a] Jan Jürjens. Composability of secrecy. In *International Workshop on Mathematical Methods, Models and Architectures for Computer Network Security (MMM-ACNS 2001)*, LNCS, St. Petersburg, 21–23 May 2001. Springer.
- [Jür01b] Jan Jürjens. Secrecy-preserving refinement. In *Formal Methods Europe (International Symposium)*, LNCS. Springer, 2001.
- [Jür01c] Jan Jürjens. Towards development of Secure systems using UMLsec. In *Fundamental Approaches to Software Engineering (FASE/ETAPS, International Conference)*, LNCS. Springer, 2001.
- [Jür01d] Jan Jürjens. Transformations for introducing patterns – a secure systems case study. In *WTUML: Workshop on Transformations in UML (ETAPS 2001 Satellite Event)*, Genova, 7 April 2001.
- [JW01] Jan Jürjens and Guido Wimmel. Security modelling for electronic commerce: The Common Electronic Purse Specifications. Submitted, 2001.
- [Low96] G. Lowe. Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR. *Software Concepts and Tools*, 17:93–102, 1996.
- [OvS94] M. Olivier and S. von Solms. A taxonomy for secure object-oriented databases. *ACM Transactions on Database Systems*, 19(1):3–46, 1994.
- [Pau98] Lawrence C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1–2):85–128, 1998.
- [RJB99] J. Rumbaugh, I. Jacobson, and G. Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 1999.
- [RSG⁺01] P. Ryan, S. Schneider, M. Goldsmith, G. Lowe, and B. Roscoe. *Modelling and Analysis of Security Protocols*. Addison Wesley, 2001. (to be published).
- [Sam00] P. Samarati. Access control: Policies, models, architectures, and mechanisms. Lecture Notes, 2000.
- [SCW00] S. Stepney, D. Cooper, and J. Woodcock. *An Electronic Purse: Specification, Refinement, and Proof*. Oxford University Computing Laboratory, 2000. Technical Monograph PRG- 126.
- [WW01] G. Wimmel and A. Wißpeitner. Extended description techniques for security engineering. In *IFIP SEC*, 2001.

This Page Intentionally Left Blank

STRONG FORWARD SECURITY

Mike Burmester*

*Department of Computer Science, Florida State University
214 Love Building, Tallahassee, Florida 32306, USA
burmester@cs.fsu.edu*

Vassilios Chrissikopoulos

*Department of Archiving and Library Studies, Ionian University
Corfu, 49100, GREECE
vchris@ionio.gr*

Panayiotis Kotzanikolaou

*Department of Informatics, University of Piraeus
80 Karaoli & Dimitriou, 185 34, Piraeus, GREECE
pktzani@unipi.gr*

Emmanouil Magkos*

*Department of Informatics, University of Piraeus
80 Karaoli & Dimitriou, 185 34, Piraeus, GREECE
emagos@unipi.gr*

Abstract

Forward security has been proposed as a method to minimize the consequences of key exposure. In this paper we analyze this method and consider a vulnerability, which is due to the fact that the exposure may not have been detected. All forward secure cryptosystems proposed so far are vulnerable during the period between key exposure and its detection. We consider the notion of strong forward security in which cryptographically processed data is protected not only for the periods prior to key exposure but also after key exposure, and present two applications with this novel property: a basic public key cryptosystem and an ElGamal-based key escrow scheme.

*Research supported by the Secretariat of Research and Technology of Greece.

Keywords: Forward security, key update, intrusion detection

1. INTRODUCTION

A major security concern in every cryptosystem is the protection of secret keys from exposure. If the adversary appropriates the secret keys of a user in an encryption scheme, then the adversary can decrypt all ciphertexts intended for that user and confidentiality is lost. For a signature scheme, the adversary can masquerade as the legitimate user.

The problem of key exposure is critical in open environments such as the Internet, where every computer node is a potential victim of hackers. Thus, there is a need to adopt mechanisms that minimize the consequences of key exposure. So far, these mechanisms generally rely on secret distributed computation [9, 14, 15, 17, 22, 29], periodical key updating and key revocation [2, 5, 11, 20, 23, 25, 27].

Gunther [20] was the first to propose an encryption key updating mechanism that protects the confidentiality of all encrypted messages prior to key exposure. With this mechanism all encrypted material is protected from key exposure after the keys are updated. This property was called *forward secrecy*. With forward secrecy, disclosure of long-term secret keying material does not compromise the secrecy of earlier encrypted material [11, 20].

A solution that establishes forward secrecy in the context of real-time multicasting over large dynamic groups was proposed by McGrew and Sherman in [27]. Burmester, Desmedt and Seberry [5] proposed an escrow system with forward secrecy. There are also solutions that address the key exposure problem for digital signatures. Herzberg *et al* [22] consider threshold signature schemes (see also [9]) in which the users update their shares proactively. These schemes offer forward security, however the distribution of shares and the distributed computation required to compute signatures make them rather inefficient (*cf.* the discussion in [2]). Bellare and Miner [2] proposed efficient digital signatures with forward security, but their security can only be proven in the Random Oracle Model [3]. Recently, Krawczyk [25] proposed a solution that can be used with any signature scheme. In this paper we shall adopt the term *forward security* both for encryption and signatures.

There is an inherent weakness in forward security that follows from the fact that the definition does not specify what happens *after an intrusion*, when the secret information has been exposed to the adversary, and *until its detection*, when the public key is revoked. During this period the security of the system is compromised. For example, suppose that the adversary (*e.g.* a hacker) has appropriated the secret keys of Alice during the session t_e but the intrusion has not been detected (Fig. 1). The adversary will be able to update the stolen keys in the same way as Alice and then generate secret keys for the sessions t_{e+1}, \dots, t_d , until the intrusion is detected. This means that cryptographically

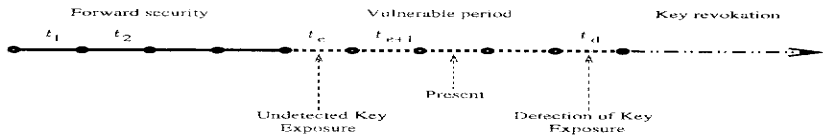


Figure 1 Forward Security

processed data after key exposure is not protected. All forward secure schemes in the literature [2,5,20,25] are vulnerable during this period. They only offer protection for sessions prior to key exposure.

Organization. In this paper we analyze forward security and consider a new threat in which the adversary appropriates *all* the secret keying material of a user without being detected. In Section 2 we consider the notion of strong forward security, in which cryptographically processed data is protected not only during the periods prior to key exposure but also during the periods after key exposure. In Section 3 we show how strong forward security can be achieved with any public key cryptosystem and in Section 4 we propose a strong forward secure key escrow/recovery scheme which is based on the ElGamal cryptosystem. We conclude in Section 5.

2. FROM FORWARD SECURITY TO STRONG FORWARD SECURITY

Suppose that Alice uses a forward secure cryptosystem and that the adversary has appropriated (all) her secret keying material during session t_e — see Figure 1. The adversary will not be able to obtain the keys for earlier sessions $t_j < t_e$, but will be able to update the key of session t_e in the same way as Alice, to get keys for sessions t_{e+1}, \dots , until session t_d , when the intrusion is detected. With the encryption scheme in [5], the updating is *deterministic* so the adversary will generate an identical key to Alice’s, and thus decrypt all ciphertexts intended for Alice. A similar argument applies to the signature schemes in [2, 25]. In this case the adversary can forge Alice’s signatures. With the encryption scheme in [20], which uses *randomized* updating, the adversary will generate a different key. However the adversary can prove that this key is “genuine”, since the adversary has also appropriated the long term authentication keys of Alice.

Regardless of whether the updating mechanism is deterministic or randomized, all cryptographically processed data is at risk during the period between

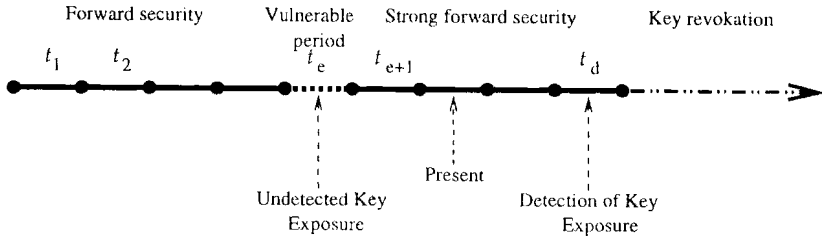


Figure 2 Strong Forward Security

key exposure and its detection. Protection from intrusions in which all the secret keying material of Alice is stolen can only be achieved by using non-cryptographic means. However, with randomized key updating this task should be easier, because Alice's updated key will be different from the key generated by the intruder (with high probability).

Definition. A system is *strongly* forward secure if disclosure of secret keying material does not compromise the security of the system for sessions both prior to exposure ($t_j < t_e$) and after exposure ($t_j > t_e$) – see Figure 2.

A practical but expensive solution. Strong forward security can be achieved with any public key cryptosystem by using threshold cryptography [9, 16, 17]. For this purpose the secret key is shared among several entities, which jointly execute the cryptographic application. The shares are then proactively updated [15, 23, 22]. Strong forward security is clearly achieved, provided that the threshold is sufficiently large.

With such schemes each application (encryption or digital signature) requires a distributed computation and therefore may be quite costly (as noted in [2]). Furthermore, the distribution of shares may be costly.

Our solution. Our goal is to achieve strong forward security in a practical and affordable way. The user must be able to certify new session keys with minimum cost, without out-of-band authentication. Furthermore, this should not involve costly distributed computations for each application (encryption or signature). For this purpose we combine randomized key updating with certification.

If a hacker appropriates the secret keying material of a legitimate user and then tries to certify an updated stolen key, then two valid public keys corresponding to the same user will be submitted for certification: the legitimate key and an alias key. The intrusion will be detected and thus the cryptographic security will only be compromised during the session of the intrusion.

3. A BASIC SOLUTION FOR ANY PUBLIC KEY CRYPTOSYSTEM

Based on our discussion above we can make any public key cryptosystem strongly forward secure. First let us consider digital signatures.

Suppose that the public/secret key pair of Alice for session t , is $(PK_{A,t}, SK_{A,t})$ and that $Cert(ID_A, PK_{A,t})$ is a certificate for it, issued by the Certifying Authority CA , where ID_A is a unique identifier of Alice. For the next session, Alice selects a random public/secret key pair $(PK_{A,t+1}, SK_{A,t+1})$, and digitally signs it together with ID_A , using her previous key: $sig_{SK_{A,t}}(ID_A, PK_{A,t+1})$. Alice then sends this together with her old certificate $Cert(ID_A, PK_{A,t})$ to the CA , which verifies Alice's signature using the old key $PK_{A,t}$. If this is correct, the CA sends Alice a new certificate $Cert(ID_A, PK_{A,t+1})$.

If an intruder appropriates (all) the secret keys of Alice during the session t (and in particular $SK_{A,t}$) and if the intruder submits an updated public key to the CA for certification, then two public keys will be submitted, both on behalf of Alice. If this happens the CA will revoke (all) the public keys of Alice.

A similar approach can be used for public key encryption. In this case however Alice needs two pairs of public keys, one for encryption and the other to authenticate her encryption key.

This basic scheme achieves strong forward security and is as secure as the underlying cryptosystem. Furthermore, it is very efficient. In particular, the certification of the public keys in each session does not require out-of-band methods. In addition, the size of keys and of the signatures does not expand as the keys are updated. However, we have a linear expansion in the number of certificates.

Remark 1. Although the protection of strong forward security is obvious in the case of encryption, one could argue that in the context of digital signatures it does not offer any additional protection to forward security. Consider for example the case when Bob has appropriated Alice's signing key. Then, even though Bob will not be able to update the stolen key without being detected, he could indirectly bypass the security of the system for future sessions. For example, he could sign postdated checks on behalf of Alice.

However, there are cases when strong forward security makes sense in the context of signatures. For example, when the lifetime of the signing key also restricts the scope of the signed message. This would make postdated checks (for later sessions) invalid.

Remark 2: "Imprisonment" attack. The proposed solution assumes that the attacker and the legitimate user have access to a Certifying Authority CA to update keys. This forces the attacker to "publish" the fact that a key has

been exposed. If the attacker can somehow prevent the legitimate user from accessing the CA, then the attacker can impersonate the user for as long as he can confine the user. There seems to be no cryptographic way to handle such attacks.

4. AN ELGAMAL KEY ESCROW SCHEME WITH STRONG FORWARD SECURITY

The solution proposed above is not satisfactory for key escrow because the updated keys must be distributed among escrow agents (an excellent survey of key escrow systems is given by Denning and Branstad in [8]). The following scheme reduces the cost of key distribution and key updating by having the escrow agents regulate the timing process for key updating.

For simplicity, we describe a basic 2-out-of-2 key escrow scheme with escrow agents EA_1 , EA_2 , a in which the Law Enforcement Agency LEA also acts as a Certifying Authority. The escrow agents and the LEA are trusted to adhere to the protocol.

Each user, say Alice, during setup, chooses a long-term secret key and shares this among the escrow agents in a verifiable way. Then, at the beginning of each session t the escrow agents select a time-control identifier h_t . This is broadcast by the LEA and will be used by all the users of the system for key updating. In particular, Alice will update her private key SK_{t-1} to SK_t by using her long-term secret key, some randomness and the time-control identifier h_t . After each updating, Alice and the escrow agents delete all information that might be useful to an adversary who may attempt to recover previous keys. Additionally, Alice updates her public key to PK_t , and proves to the LEA in zero-knowledge [19] that this has been properly constructed. The LEA then certifies the updated public key PK_t .

A hacker who succeeds in appropriating Alice's secret keying material may attempt to update the stolen session key and to get the updated key certified by the LEA. However, Alice will also submit her updated key for certification. The two keys are different (with overwhelming probability). The LEA will notice that different keys corresponding to the same user are submitted for certification, and thus detect the intrusion and revoke all the public keys of Alice.

Background. We use an ElGamal encryption scheme [12]. Let r , q , p be large primes with $q = 2r + 1$, $p = 2q + 1$, and let H be a subgroup of Z_q^* of order r with generator h , and G be a subgroup of Z_p^* of order q with generator g . For simplicity, and when there is no ambiguity, we drop the modulus operators. Also, we write $a \in_R A$ to indicate that the element a is chosen randomly with uniform distribution from the set A .

The Diffie-Hellman [10] operator DH is defined by $DH(g^a, g^b) = g^{ab}$. Given the numbers g^a and g^b , the problem of computing $DH(g^a, g^b)$ is called

the *Diffie-Hellman* problem. The problem of deciding whether $z = DH(g^a, g^b)$, for a given $z \in Z_p$, is called the *Decision Diffie-Hellman* DDH problem [10].

Setup. Alice chooses a long term private key $x_A \in_R Z_q^*$ and computes $y_A = g^{x_A}$. Alice gives her long term public key $PK_A = \langle p, q, g, y_A \rangle$ to the LEA, authenticates it by non-cryptographic (out-of-band) means, and gets a certificate $Cert(ID_A, PK_A)$. Then,

- 1 Alice chooses shares $x_1 \in_R Z_q^*$ and $x_2 = x_A(x_1)^{-1}$. Alice gives the shares x_1, x_2 privately to the escrow agents EA_1, EA_2 , respectively.
- 2 The escrow agents check that $y_A = DH(g^{x_1}, g^{x_2})$. If not, Alice is reported to the LEA.

Key updating (session $t = 1, 2, \dots$). Agents EA_1, EA_2 choose numbers $r_{1,t}, r_{2,t} \in_R Z_r^*$ respectively, and jointly construct $h^{r_t} = h^{r_{1,t}r_{2,t}}$ in a secure way by using the Diffie-Hellman key exchange protocol [10]. The agents send h^{r_t} to the LEA which publishes it. This number identifies the session t , and is used by *all* the users of the system. It represents the randomness of the escrow agents in the key updating procedure and is the same for all users. The agents then discard the exponents $r_{1,t-1}$ and $r_{2,t-1}$ of the previous session (when $t > 1$). Then:

- 1 Alice chooses a number $r_{A,t} \in_R Z_r^*$, computes $h^{r_{A,t}}$ and sends this to the LEA. Alice also computes the Diffie-Hellman key $h_t = h^{r_t r_{A,t}}$.
- 2 Alice updates her secret key for session t to $SK_{A,t} = h_t x_A$. She then computes $y_{A,t} = g^{h_t x_A}$, and sends to the LEA her public session key $PK_{A,t} = \langle p, q, r, g, h, y_{A,t} \rangle$. Alice then proves in zero knowledge (see the Appendix) that $y_{A,t} = g^{DH(h^{r_t}, h^{r_{A,t}})DL(g^{x_A})}$, where $DL(g^{x_A})$ is the discrete logarithm of g^{x_A} . If the proof is correct, the LEA certifies the updated public key and issues Alice with a certificate $Cert(ID_A, PK_{A,t})$. Then Alice discards $r_{A,t}$ and the previous session key.

Getting an escrowed key. Assume that a court order has been issued to decrypt all ciphertexts intended for Alice during session t . Then the LEA will wiretap the communication of Alice. Let $(g^k, m(y_{A,t})^k)$ be an ElGamal encryption of a message m sent to Alice during this session. The LEA will send g^k and $h^{r_{A,t}}$ to the escrow agents. The agents first compute the Diffie-Hellman key $h_t = h^{r_t r_{A,t}}$, and then the factor $(y_{A,t})^k = (((g^k)^{h_t})^{x_1})^{x_2}$. They send $(y_{A,t})^k$ to the LEA for decryption.

Theorem 1 *If the Decision Diffie-Hellman problem is hard then the proposed escrow scheme has strong forward security.*

Proof. Suppose that there is a polynomial time algorithm A that breaks the proposed escrow scheme. Let $z, h^a, h^b \in_R Z_q^*$ be an input for the Decision Diffie-Hellman problem. We shall use A to break the DDH problem.

Choose at random a secret key $x_A \in_R Z_q^*$ and let $y_A = g^{x_A}$ be the long-term public key. Next, prepare a history of ciphertext-message pairs (c, m) for A , for earlier sessions j , by choosing at random $k \in_R Z_q^*$, $m \in_R Z_p^*$ and $r_j, r_{A,j} \in_R Z_r^*$ and take $c = (g^k, mg^{kx_A}h^{r_j r_{A,j}})$.

Give to A : x_A, y_A , and z, h^a, h^b instead of $h_t, h^{r_{A,t}}, h^{r_t}$, the public (session) key $y_{A,t} = g^{z x_A}$, a history of ciphertext-message pairs and the “ciphertext”:

Let the output of A be m' . If $m' = m$ then the decision is that $z = h^{ab}$, else $z \neq h^{ab}$.

Remark 3. The interactive zero knowledge proof in Step 2 of the key updating can be replaced by a signature, using the Fiat–Shamir heuristic [13], which requires a hash function. However it should be noted that if we use such signatures then the security of the scheme can only be proven in the Random Oracle Model [3].

Remark 4. In Section 2 we considered a solution involving the distribution of the secret keys via secret sharing in a proactive way. In our protocol above we also distribute the keys and use an updating mechanism similar to proactive mechanisms. However, our encryptions do not require a distributed computation.

Remark 5. The escrow agents are safe repositories for the long-term secret keys of all the users of the system. In our protocol the agents also generate a random number h^{r_t} . This number is for a specific time period and is the same for *all the users of the system*. In the next session a new random number is chosen and the old one is discarded. Observe that the addition or the removal of a user from the system does not affect the functionality of the agents.

Remark 6. The ElGamal escrow scheme described above can easily be modified to get a Key Recovery scheme by replacing the LEA and the escrow agents with a Data Recovery Agency and recovery agents respectively. Observe that if the keys to be recovered encrypt archived data, then there is no point in adopting a Key Recovery scheme with forward secrecy, as observed in [1]. Consequently, the proposed scheme can only be used to recover encrypted traffic.

Generalizations

- 1 It is easy to see how to generalize this scheme to a t -out-of- l key escrow scheme. Robustness can be achieved by using the approach in [16, 17].

Furthermore, our scheme can be easily modified to prevent subliminal channel attacks, as described in [24].

- 2 It is well known that the ElGamal encryption scheme is not semantically secure [18]. To extend our scheme to a semantically secure scheme we can use the Cramer-Shoup extension of ElGamal [7].

5. CONCLUSION

Forward security protects cryptographically processed data prior to key exposure. However in many applications it is difficult to detect intrusions. Indeed, hackers will not necessarily use the appropriated keys until this is expedient or profitable. It is therefore important to consider mechanisms, which also protect cryptographically processed data after an intrusion. Strong forward security offers such protection.

References

- [1] H. Abelson, R. Anderson, S. Bellare, J. Benaloh, M. Blaze, W. Diffie, J. Gilmore, P. Neumann, R. Rivest, J. Schiller, B. Schneier. *The Risks of Key Recovery, Key Escrow, Trusted Third Party & Encryption*, Digital Issues, No. 3, 1998, pp. 1–18.
- [2] M. Bellare, S. Miner. *A Forward-Secure Digital Signature Scheme*, Advances in Cryptology — Crypto '99, Proceedings, Lecture Notes in Computer Science, Vol. 1666, Springer 1999, pp. 197–207.
- [3] M. Bellare, P. Rogaway. *Random Oracles are Practical*, 1st Annual Conference on Computer and Communications Security, Proceedings, ACM 1993, pp. 154–164.
- [4] D. Boneh. *The Decision Diffie-Hellman Problem*, 3rd Algorithmic Number Theory Symposium, Proceedings, Lecture Notes in Computer Science, Vol. 1423, Springer 1998, pp. 48–63.
- [5] M. Burmester, Y. Desmedt, J. Seberry. *Equitable Key Escrow with Limited Time Span (or How to Enforce Time Expiration Cryptographically)*, Advances in Cryptology — AsiaCrypt '98, Proceedings, Lecture Notes in Computer Science Vol. 1514, Springer 1998, pp. 380–391.
- [6] D. Chaum. *Zero-Knowledge Undeniable Signatures*, Advances in Cryptology — Eurocrypt '90, Proceedings, Lecture Notes in Computer Science, Vol. 473, Springer 1991, pp. 458–464.
- [7] R. Cramer, V. Shoup. *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack*, Advances in Cryptology — Crypto '98, Proceedings, Lecture Notes in Computer Science, Vol. 1462, Springer 1998, pp. 13–25.

- [8] D.E. Denning and D.K. Branstad. *A taxonomy of Key Escrow Encryption Systems*, Communications of the ACM, Vol. 39(3), 1996, pp. 24–40.
- [9] Y. Desmedt, Y. Frankel. *Threshold Cryptosystems*, Advances in Cryptology — Crypto '89, Proceedings, Lecture Notes in Computer Science, Vol. 435, Springer 1989, pp. 307–315.
- [10] W. Diffie, M. Hellman. *New Directions in Cryptography*, IEEE Transactions on Information Theory, Vol. 22(6), IEEE 1976, pp. 644–654.
- [11] W. Diffie, P. Van Oorschot and M. Wiener. *Authentication and Authenticated Key Exchanges*, Designs, Codes and Cryptography, Vol. 2, 1992, pp. 107–125.
- [12] T. ElGamal. *A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms*, IEEE Transactions on Information Theory, Vol. 31, IEEE 1985, pp. 469–472.
- [13] A. Fiat, A. Shamir. *How to prove yourself: Practical Solutions to Identification and Signature Problems*, Advances in Cryptology — Crypto '86, Proceedings, Lecture Notes in Computer Science, Vol. 263, Springer 1986, pp. 186–194.
- [14] Y. Frankel, P. Gemmel, P.D. MacKenzie, and M. Yung. *Proactive RSA*, Advances in Cryptology — Crypto '97, Proceedings, Lecture Notes in Computer Science, Vol. 1109, Springer 1997, pp. 440–454.
- [15] Y. Frankel, P. Gemmel, P. D. MacKenzie and M. Yung. *Optimal-Resilience Proactive Public-Key Cryptosystems*, 38th Annual Symp. on Foundations of Computer Science, Proceedings, IEEE 1997, pp. 384–393.
- [16] Y. Frankel, P. Gemmel, and M. Yung. *Witness Based Cryptographic Program Checking and Robust Function Sharing*, 28th annual ACM Symp. Theory of Computing, Proceedings, ACM 1996, pp. 499–508.
- [17] R. Gennaro, S. Jarecki, H. Krawczyk and T. Rabin. *Robust and Efficient Sharing of RSA Functions*, Advances in Cryptology — Crypto '96, Proceedings, Lecture Notes in Computer Science, Vol. 1109, Springer 1996, pp. 157–172.
- [18] S. Goldwasser, S. Micali. *Probabilistic Encryption*, Journal of Computer and System Sciences, Vol. 28, 1984, pp. 270–299.
- [19] S. Goldwasser, S. Micali, and C. Rackoff. *The Knowledge Complexity of Interactive Proof Systems* Proceedings of the 17th ACM Symposium on the Theory of Computing STOC, ACM Press, Providence, Rhode Island, U.S.A., 1985, pp. 291–304.
- [20] C. Gunther. *An Identity-based Key-Exchange Protocol*, Advances in Cryptology — Eurocrypt '89, Proceedings, Lecture Notes in Computer Science, Vol. 434, Springer 1989, pp. 29–37.

- [21] S. Haber and W. Storenetta. *How to Timestamp a Document*, Advances in Cryptology — Crypto '90, Proceedings, Lecture Notes in Computer Science, Vol. 537, Springer 1990.
- [22] A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, M. Yung. *Proactive Public Key and Signature Schemes*, 4th Annual Conference on Computer and Communications Security, Proceedings, ACM 1997, pp. 100–110.
- [23] A. Herzberg, S. Jarecki, S. Krawczyk, and M. Yung. *Proactive Secret Sharing*, Advances in Cryptology — Crypto '95, Proceedings, Lecture Notes in Computer Science, Vol. 963, Springer 1995, pp. 339–352.
- [24] J. Kilian, T. Leighton. *Fair Cryptosystems, Revisited*, Advances in Cryptology — Eurocrypt '95, Proceedings, Lecture Notes in Computer Science, Vol. 963, Springer 1995, pp. 208–220.
- [25] H. Krawczyk. *Simple Forward-Secure Signatures For Any Signature Scheme*, Proceedings of the 7th ACM Conference on Computer and Communications Security, ACM Press, 2000, pp. 108–115.
- [26] U. Maurer, Y. Wolf. *Diffie-Hellman Oracles*, Advances in Cryptology — Crypto '96, Proceedings, Lecture Notes in Computer Science, Vol. 1109, Springer 1996, pp. 268–282.
- [27] D. McGrew, A. Sherman. *Key Establishment in Large Dynamic Groups*, Manuscript, Submitted to IEEE Transactions on Software Engineering.
- [28] A. Menezes, P. Van Oorschot, S. Vanstone. *Handbook of Applied Cryptography*, CRC Press, 1997.
- [29] T. Rabin. *A Simplified Approach to Threshold and Proactive RSA*, Advances in Cryptology — Crypto '98, Proceedings, Lecture Notes in Computer Science, Vol. 1462, Springer 1997, pp. 89–104.
- [30] A. Shamir. *How to Share a Secret*, Communications of the ACM, Vol. 22, ACM 1979, pp. 612–613.

Appendix

Let

$$\mathcal{L} = \{(p, q, r, g, g^a, h^b, h^c, z) \mid p, q, r \text{ primes, } p = 2q + 1, q = 2r + 1, \\ g \text{ a generator of } Z_p^*, h \text{ a generator of } Z_q^*, a \in Z_q^*, b, c \in Z_r^*, \text{ and } \\ z \in Z_p^* \text{ with } z = g^{ah^{bc}} \pmod{p}\}.$$

An interactive zero-knowledge proof of membership in L

Input: $x = (p, q, r, g, g^a, h^b, h^c, z)$

Repeat l times ($l = \Theta(\log p)$):

- 1 The Prover chooses $k \in_R Z_q^*$, $t \in_R Z_r^*$, computes $u = ka \pmod{q}$, $v = c + t \pmod{r}$, and then sends to the Verifier:

$$X = g^{uh^{bv}}, Y = g^u, Z = h^v.$$

- 2 The Verifier sends to the Prover a bit query $e \in \{0, 1\}$.

- 3 The Prover sends to the Verifier:

$$\begin{array}{ll} (u, v), & \text{if } e = 0 \\ (k, t), & \text{if } e = 1. \end{array}$$

Verification: The Verifier checks that:

$$\begin{array}{ll} \text{when } e = 0, & X = g^{u(h^b)^v}, \quad Y = g^u, \quad Z = h^v \\ \text{when } e = 1, & X = z^{k(h^b)^t}, \quad Y = (g^a)^k, \quad Z = h^c h^t. \end{array}$$

The Verifier accepts (that $x \in L$) if the verification is satisfied for all k rounds.

Proof of correctness

Completeness: If $x \in L$ then the Verifier will always accept.

Soundness: If the Verifier accepts with non-negligible probability ($\geq 1/\text{poly}(\log p)$), then the Prover must answer correctly both queries $e = 0$, $e = 1$ for some triple X, Y, Z . Therefore,

$$\begin{array}{ll} Z = h^v = h^c h^t & \Rightarrow v = c + t \pmod{r} \\ Y = g^u = (g^a)^k & \Rightarrow u = ka \pmod{q} \\ X = g^{u(h^b)^v} = g^{ka h^{b(c+t)}} = z^{k(h^b)^t} & \Rightarrow z = g^{ah^{bc}}. \end{array}$$

It follows that $x \in L$.

Simulation (zero-knowledge):

- when $e = 0$, choose random u, v and construct X, Y, Z as in Step 1;
- when $e = 1$, choose random k, t and construct $X = z^{k(h^b)^t}$, $Y = (g^a)^k$,
and $Z = h^c h^t$.

This Page Intentionally Left Blank

SECRET SHARING AND VISUAL CRYPTOGRAPHY SCHEMES

A Randomness Preserving Transformation

Annalisa De Bonis and Alfredo De Santis

Dipartimento di Informatica ed Applicazioni

Università di Salerno, 84081 Baronissi (SA), Italy

{debonis,ads}@dia.unisa.it

Abstract

A secret sharing scheme is a method for sharing a secret among a set P of n participants. The secret is encoded into n pieces called shares each of which is given to a distinct participant. Certain *qualified* subsets of participants can recover the secret by pooling together their information, whereas *forbidden* subsets of participants have no information on the secret. The specification of the qualified sets and the forbidden sets is called *access structure*.

A special kind of secret sharing schemes are visual cryptography schemes (VCSs), that is, schemes where the secret to share is an image and the shares consist of xeroxed transparencies which are stacked to recover the shared image.

In this paper we analyze the relationship between secret sharing schemes and VCSs, focusing our attention on the amount of randomness required to generate the shares. We show how to transform a secret sharing scheme for a given access structure into a VCS for the same access structure while preserving the randomness of the original scheme. An important consequence of this transformation is that lower bounds on the randomness of visual cryptography schemes apply to general secret sharing schemes. Our randomness preserving transformation has also been applied to derive a new upper bound on the randomness of (k, n) -threshold VCSs which dramatically improves on the previously known bounds. All VCSs obtained by applying our randomness preserving transformation allow a perfect reconstruction of black pixels.

Keywords: Cryptography, Randomness, Secret Sharing, Visual Cryptography.

Introduction

A secret sharing scheme is a method for sharing a secret among a set P of n participants. The secret is encoded into n pieces called shares each of

which is given to a distinct participant. Certain *qualified* subsets of participants can recover the secret by pooling together their information, whereas *forbidden* subsets of participants have no information on the secret. The specification of all qualified and forbidden subsets of participants constitutes an *access structure*.

Secret sharing schemes are especially useful in situations which require that several people cooperate in order to start an important action such as opening a bank vault or a safety deposit box, or launching a missile.

Shamir [14] and Blakley [5] have been the first to introduce secret sharing schemes. In particular, they considered (k, n) -threshold schemes, that is scheme where only subsets of P of size larger than or equal to a fixed integer k can reconstruct the secret. Ito, Saito, and Nishizeki [11] showed how to realize a secret sharing scheme for any access structure. Later, Benaloh and Leichter [4] proposed a simpler and more efficient way to realize secret sharing schemes. Other general techniques handling arbitrary access structures can be found in [12, 17].

An important issue in the implementation of secret sharing schemes is the amount of randomness required for generating the shares. Blundo *et al.* [7] have been the first to analyze the randomness of secret sharing schemes. Random bits are a natural computational resource which must be taken into account when designing cryptographic algorithms. Considerable effort has been devoted to reduce the number of bits used by probabilistic algorithms (see for example [10]) and to analyze the amount of randomness required in order to achieve a given performance. Motivated by the fact that “truly” random bits are hard to generate, it has also been investigated the possibility of using imperfect source of randomness in randomized algorithms [19]. In spite of the considerable effort devoted to analyzing the incidence of randomness in several areas of computer science, very few results have been obtained to quantify the amount of random bits required to solve classes of problems.

A special kind of secret sharing schemes are *visual cryptography schemes*. A visual cryptography scheme (VCS) is a method to secretly share an image among a given group of participants. A VCS for a set P of n participants encodes a secret image into n shadow images which constitute the shares given to the n participants. The shares given to participants in $X \subseteq P$ are xeroxed onto transparencies. If X is *qualified* then the participants in X can visually recover the secret image by stacking their transparencies without any cryptography knowledge and without performing any cryptographic computation.

In this paper we analyze the relationship between secret sharing schemes and visual cryptography schemes, with a special concern for the amount of randomness required to generate the shares. In this paper we only consider VCSs for black and white images. Visual cryptography schemes for black and white images have been defined by Naor and Shamir in [13]. They analyzed (k, n) -threshold visual cryptography schemes. Ateniese *at al.* [1,2] extended

the model by Naor and Shamir to general access structures. Since in a VCS an image is encoded pixel by pixel, then a VCS for black and white images is a special case of secret sharing scheme for a set of secrets of size two. We refer to such a secret sharing scheme with the term of *Binary Secret Sharing Scheme* (BSS). It follows that lower bounds on the randomness of BSSs apply also to VCSs. In this paper we prove that the converse implication holds as well, thus shading a new light on the study of secret sharing schemes. In other words, we prove that the number of random bits needed to secretly share a pixel is the same as that needed to share any secret chosen in a set of size two. Indeed, given a BSS Σ for an access structure Γ , we show how to construct a VCS for Γ with the same randomness as Σ . Such construction technique will be also applied to derive a new upper bound on the randomness of (k, n) -threshold VCSs. This upper bound dramatically improves on all previously known upper bounds and it is very close to the best known lower bound [9].

1. THE MODEL

Let $P = \{1, \dots, n\}$ be a set of elements called *participants*, and let 2^P denote the set of all subsets of P . Let $\Gamma_{\text{Qual}} \subseteq 2^P$ and $\Gamma_{\text{Forb}} \subseteq 2^P$, where $\Gamma_{\text{Qual}} \cap \Gamma_{\text{Forb}} = \emptyset$. We refer to members Γ_{Qual} as *qualified sets* and we call members of Γ_{Forb} *forbidden sets*. The pair $\Gamma = (\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$ is called the *access structure* of the scheme.

Let Γ_0 consist of all the minimal qualified sets:

$$\Gamma_0 = \{A \in \Gamma_{\text{Qual}} : A' \notin \Gamma_{\text{Qual}} \text{ for all } A' \subset A\}.$$

A participant $p \in P$ is an *essential* participant if there exists a set $X \subseteq P$ such that $X \cup \{p\} \in \Gamma_{\text{Qual}}$ but $X \notin \Gamma_{\text{Qual}}$. A non-essential participant does not need to participate “actively” in the reconstruction of the secret, since the information she has is not needed by any set in P in order to recover the shared image. In any secret sharing scheme having non-essential participants, these participants do not require any information in their shares.

In the case where Γ_{Qual} is monotone increasing, Γ_{Forb} is monotone decreasing, and $\Gamma_{\text{Qual}} \cup \Gamma_{\text{Forb}} = 2^P$, the access structure is said to be *strong*, and Γ_0 is termed a *basis*. In a strong access structure,

$$\Gamma_{\text{Qual}} = \{C \subseteq P : B \subseteq C \text{ for some } B \in \Gamma_0\},$$

and we say that Γ_{Qual} is the *closure* of Γ_0 .

In the following we formally define secret sharing schemes for a strong access structure $(\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$. Indeed, in traditional secret sharing schemes the access structures are always assumed to be strong.

A secret sharing scheme Σ for a set of secrets $S = \{s_0, \dots, s_{h-1}\}$ on a set P of participants for the strong access structure $(\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$ is a method to

secretly share a secret chosen in S among the members of P in such a way that only subsets of participants which are in Γ_{Qual} can recover the secret. The secret sharing scheme Σ consists of h collections of *distribution functions* C_0, \dots, C_{h-1} . A *distribution function* $f \in C_i, i = 0, \dots, h - 1$, is a function which associates to each participant $p \in P$ a share. When the secret to share is $s_i, i = 0, \dots, h - 1$, the dealer randomly chooses a distribution function $f \in C_i$ and assigns to each $p \in P$ the share $f(p)$.

Definition 1 Let $(\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$ be a strong access structure on a set P of participants. The collections of distribution functions C_0, \dots, C_{h-1} realize a secret sharing scheme for a set of secrets of size h if the following conditions hold:

1. Any subset $X \subseteq P$ of participants qualified to recover the secret can compute the secret.

Formally, if $X \in \Gamma_{\text{Qual}}$, then it is $\{(p, f(p))\}_{p \in X} \neq \{(p, g(p))\}_{p \in X}$, for all $f \in C_i$ and $g \in C_j$ with $i, j \in \{0, \dots, h - 1\}$ and $i \neq j$.

2. Any subset $X \subseteq P$ of participants non-qualified to recover the secret has no information on the secret value.

Formally, if $X = \{p_{v_1}, \dots, p_{v_a}\} \in \Gamma_{\text{Forb}}$, then for any possible choice $sh_{v_1} \dots, sh_{v_a}$ of the shares given to participants p_{v_1}, \dots, p_{v_a} , it results

$$\frac{|\{f \in C_i : (f(p_{v_1}), \dots, f(p_{v_a})) = (sh_{v_1}, \dots, sh_{v_a})\}|}{|C_i|} = \frac{|\{f \in C_j : (f(p_{v_1}), \dots, f(p_{v_a})) = (sh_{v_1}, \dots, sh_{v_a})\}|}{|C_j|},$$

for any $i, j \in \{0, \dots, h - 1\}$.

The first property is related to the reconstruction of the secret. It states that the for any pair of distinct secrets s_i and s_j , the group of shares assigned to a qualified group of participants when the encoded secret is s_i is different from that assigned to the same group of participants when the encoded secret is s_j .

The second property is called *security*, since it implies that, even by inspecting all their shares, a forbidden set of participants cannot gain any information on the shared secret.

Notice that in the previous definition $C_i, i = 0, \dots, h - 1$, is a multiset of distribution functions, therefore we allow a function to appear more than once in $C_i, i = 0, \dots, h - 1$. Moreover, the sizes of the collections C_0, \dots, C_{h-1} do not need to be the same.

The *randomness* of a secret sharing scheme represents the number of random bits used by the dealer to share a secret among the participants. Let Σ be a secret sharing scheme for a set of h secrets s_0, \dots, s_{h-1} realized by the collections C_0, \dots, C_{h-1} . For $i = 0, \dots, h - 1$, let p_i denote the probability that the shared secret is s_i . The randomness of Σ has been defined by Blundo *et al.* [7] as

$$\mathcal{R}^{(C_0, \dots, C_{h-1}); \mathbf{P}} = \sum_{i=0}^{h-1} p_i \log |C_i|,$$

where $\mathbf{p} = (p_0, \dots, p_{h-1})$. Let $\Gamma = (\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$ be a given access structure. In accordance with [7], the dealer's randomness for the access structure Γ is defined as

$$\mathcal{R}_\Gamma = \inf_{\mathcal{A}, \mathcal{I}} \mathcal{R}^{(C_0, \dots, C_{h-1}), \mathbf{p}},$$

where A denotes the set of all h -tuple of collections C_0, \dots, C_{h-1} realizing a secret sharing scheme for Γ for the set of secrets $\{s_0, \dots, s_{h-1}\}$, and I is the set of all probability vectors of length h with non-zero entries. Indeed, we assume that the secret have non-zero probability of being any of s_0, \dots, s_{h-1} . In [7] the above definition has been proved to be equivalent to the following

$$\mathcal{R}_\Gamma = \min_A \log(\min\{|C_0|, \dots, |C_{h-1}|\}).$$

The above definition implies that, given h function collections C_0, \dots, C_{h-1} realizing a secret sharing scheme for a set of h secrets for the access structure Γ , we are mainly concerned with the quantity $\log(\min\{|C_0|, \dots, |C_{h-1}|\})$. Hence, we define the randomness $R(C_0, \dots, C_{h-1})$ of a secret sharing scheme for a set of h secrets realized by C_0, \dots, C_{h-1} as

$$\mathcal{R}(C_0, \dots, C_{h-1}) = \log(\min\{|C_0|, \dots, |C_{h-1}|\}). \quad (1)$$

1.1. VISUAL CRYPTOGRAPHY SCHEMES

We assume that the image to be encoded consists of a collection of black and white pixels. The image is encoded pixel by pixel. A pixel is encoded into n pixels which constitute the shares for the n participants associated with that pixel. For each participant the shares associated with the pixels of the whole secret image are xeroxed onto a transparency which constitutes the share assigned to that participant. The participants of a qualified set can visually recover the secret image by stacking their transparencies.

As an example, consider the image representing the acronym "SEC2001".

SEC2001

The two shares generated by a (2, 2)-threshold VCS are given below.

Share of participant 1



Share of participant 2



The following is the image obtained by stacking the shares of both participants



Each of the n shares associated with a single pixel is a collection of m black and white subpixels. The resulting structure can be described by an $n \times m$ boolean matrix $S = [s_{ij}]$ where $s_{ij} = 1$ iff the j -th subpixel in the i -th transparency is black. Therefore the grey level of the combined shares, obtained by stacking the transparencies i_1, \dots, i_s is proportional to the Hamming weight $w(V)$ of the m -entry vector $V = OR(R_{i_1}, \dots, R_{i_s})$, where R_{i_1}, \dots, R_{i_s} are the rows of S associated with the transparencies we stack. This grey level is interpreted by the visual system of the users as black or as white according with some rule of contrast.

Definition 2 Let $(\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$ be an access structure on a set of n participants. Two collections (multisets) of $n \times m$ boolean matrices $\hat{\mathcal{C}}_0$ and $\hat{\mathcal{C}}_1$ constitute a visual cryptography scheme $(\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$ -VCS if there exist a value $\alpha(m)$ and a collection $\{(X, t_X)\}_{X \in \Gamma_{\text{Qual}}}$ satisfying:

1 Any (qualified) set $X = \{i_1, i_2, \dots, i_p\} \in \Gamma_{\text{Qual}}$ can recover the shared image by stacking their transparencies.

Formally, for any $M \in \hat{\mathcal{C}}_0$, the “or” V of rows i_1, i_2, \dots, i_p satisfies $w(V) \leq t_X - \alpha(m) \cdot m$; whereas, for any $M \in \hat{\mathcal{C}}_1$ it results that $w(V) \geq t_X$.

2 Any (forbidden) set $X = \{i_1, i_2, \dots, i_p\} \in \Gamma_{\text{Forb}}$ has no information on the shared image.

Formally, the two collections of $p \times m$ matrices obtained by restricting the $n \times m$ matrices of $\hat{\mathcal{C}}_0$ and $\hat{\mathcal{C}}_1$ to rows i_1, i_2, \dots, i_p are indistinguishable, in the sense that they contain the same matrices with the same frequencies.

Each pixel of the original image will be encoded into n pixels, each of which consists of m subpixels. To share a white (black, resp.) pixel, the dealer randomly chooses one of the matrices in $\hat{\mathcal{C}}_0$ ($\hat{\mathcal{C}}_1$ resp.) and distributes row i to participant i .

The first property of Definition 2 is related to the contrast of the image. It states that when a qualified set of users stack their transparencies they can correctly recover the shared image. Observe that this property implies Property 1. of Definition 1. The value $\alpha(m)$ is called *relative difference*, the number $\alpha(m) \cdot m$ is referred to as the *contrast* of the image, the set $\{(X, t_X)\}_{X \in \Gamma_{\text{Qual}}}$ is called the *set of thresholds*, and t_X is the threshold associated to $X \in \Gamma_{\text{Qual}}$. We want the contrast to be as large as possible and at least one, that is, $\alpha(m) \geq 1/m$. The second property, as well as Property 2. of Definition 1, is related to the security of the scheme.

The model of visual cryptography we consider is the same as that described in [1,2]. This model is a generalization of the one proposed in [13], since with each set $X \in \Gamma_{\text{Qual}}$ we associate a (possibly) different threshold t_X . Further, the access structure is not required to be strong in our model.

Notice that if a set of participants X is a superset of a qualified set X' , then they can recover the shared image by considering only the shares of the set X' . This does not in itself rule out the possibility that stacking all the transparencies of the participants in X does not reveal any information about the shared image.

In accordance with definition (1), the randomness $R(\hat{\mathcal{C}}_0, \hat{\mathcal{C}}_1)$ of a visual cryptography scheme realized by $\hat{\mathcal{C}}_0$ and $\hat{\mathcal{C}}_1$ is given by

$$\mathcal{R}(\hat{\mathcal{C}}_0, \hat{\mathcal{C}}_1) = \log(\min\{|\hat{\mathcal{C}}_0|, |\hat{\mathcal{C}}_1|\}).$$

The randomness of a VCS represents the number of random bits per pixel required by the VCS to share a secret image.

2. A RANDOMNESS PRESERVING TRANSFORMATION FROM BSSs TO VCSs

In this section we will show how to transform a BSS for a strong access structure Γ into a VCS for Γ with the same randomness as the original BSS.

Let $\mathcal{C}_0 = \{f_1^0, \dots, f_{c_0}^0\}$ and $\mathcal{C}_1 = \{f_1^1, \dots, f_{c_1}^1\}$ be two function collections realizing a BSS for an access structure on the set of participants $P = \{1, \dots, n\}$. Two tables, T_0 and T_1 , will be used to represent the shares assigned to each participant by the distribution functions of \mathcal{C}_0 and \mathcal{C}_1 . For any $b \in \{0, 1\}$, $i = 1, \dots, n$ and $j = 1, \dots, c_b$, it is $T_b[i, j] = f_j^b(i)$. A share will be symbolically represented by a literal indexed with the associated participant. For a given participant $i \in \{1, \dots, n\}$, distinct literals indexed with i denote distinct shares. Notice that Property 1. of Definition 1 implies that if we restrict T_0 and T_1 to

the rows corresponding to a set $X \in \Gamma_{\text{Qual}}$, we obtain two tables having no common column. Moreover, Property 2. of Definition 1 implies that if we restrict T_0 and T_1 to the rows corresponding to a set $X \in \Gamma_{\text{Forb}}$, we obtain two tables whose multisets of columns are indistinguishable, in the sense that they contain the same columns with the same frequencies.

The following example illustrates the randomness preserving transformation. For any n -row matrix M and any set $X \subseteq \{1, \dots, n\}$, we will denote with $M[X]$ the matrix obtained by restricting M to the rows with indices in X . The rows appear in $M[X]$ in the same order they appear in M .

The initial BSS

Let us consider the strong access structure Γ on the set of participants $\{1, 2, 3, 4\}$ with basis $\Gamma_0 = \{\{1, 3, 4\}, \{1, 2\}, \{2, 3\}, \{2, 4\}\}$. Let us assume that $C_0 = \{f_1^0, f_2^0, f_3^0, f_4^0\}$ and $C_1 = \{f_1^1, f_2^1, f_3^1, f_4^1\}$ be two collections of distribution functions realizing a BSS for Γ and that the shares assigned to each participant by the distribution functions of C_0 and C_1 be given by the following two tables

	f_1^0	f_2^0	f_3^0	f_4^0	
$T_0 =$	1	x_1	x_1	y_1	y_1
	2	x_2	y_2	z_2	w_2
	3	x_3	y_3	x_3	y_3
	4	x_4	y_4	y_4	x_4

	f_1^1	f_2^1	f_3^1	f_4^1	
$T_1 =$	1	x_1	y_1	y_1	x_1
	2	w_2	x_2	y_2	z_2
	3	x_3	y_3	x_3	y_3
	4	y_4	y_4	x_4	x_4

Construction of the Matrix collections \hat{C}_0 and \hat{C}_1

We associate to each function $f_j^b, j = 1, 2, 3, 4$ and $b \in \{0, 1\}$, a 4×4 matrix M_j^b . For $j = 1, 2, 3, 4$, and $b = 0, 1$, we construct the matrix M_j^b as follows. For any $i = 1, 2, 3, 4$ and $l = 1, 2, 3, 4$, we set the i -th entry of the l -th column of M_j^b equal to

$$M_j^b[i, \ell] = \begin{cases} 0 & \text{if } f_j^b(i) = f_\ell^0(i), \\ 1 & \text{otherwise.} \end{cases}$$

The matrices resulting from the above construction for our running example are:

$$M_1^0 = \begin{bmatrix} 0011 \\ 0111 \\ 0101 \\ 0110 \end{bmatrix} \quad M_2^0 = \begin{bmatrix} 0011 \\ 1011 \\ 1010 \\ 1001 \end{bmatrix} \quad M_3^0 = \begin{bmatrix} 1100 \\ 1101 \\ 0101 \\ 1001 \end{bmatrix} \quad M_4^0 = \begin{bmatrix} 1100 \\ 1110 \\ 1010 \\ 0110 \end{bmatrix}$$

$$M_1^1 = \begin{bmatrix} 0011 \\ 1110 \\ 0101 \\ 1001 \end{bmatrix} \quad M_2^1 = \begin{bmatrix} 1100 \\ 0111 \\ 1010 \\ 1001 \end{bmatrix} \quad M_3^1 = \begin{bmatrix} 1100 \\ 1011 \\ 0101 \\ 0110 \end{bmatrix} \quad M_4^1 = \begin{bmatrix} 0011 \\ 1101 \\ 1010 \\ 0110 \end{bmatrix}$$

The reader can quickly verify by a simple inspection of the collections $\{M_1^0, M_2^0, M_3^0, M_4^0\}$ and $\{M_1^1, M_2^1, M_3^1, M_4^1\}$ that the above construction yields a VCS for the access structure $(\Gamma_{\text{Qual}}, \Gamma_{\text{Forb}})$.

In the following we describe an algorithm which transforms an arbitrary BSS for a given access structure into a VCS for the same access structure. Let $C_0 = \{f_1^0, \dots, f_{c_0}^0\}$ and $C_1 = \{f_1^1, \dots, f_{c_1}^1\}$ be two collections of distribution functions realizing a BSS for a given strong access structure Γ . The input of the algorithm consists of the two tables T_0 and T_1 representing the shares assigned to each participant by the distribution functions of C_0 and C_1 .

```

Generate-VCS( $T_0, T_1$ )
   $n \leftarrow$  number of rows of  $T_0$ 
   $c_0 \leftarrow$  number of columns of  $T_0$ 
   $c_1 \leftarrow$  number of columns of  $T_1$ 
  for  $b \leftarrow 0$  to 1
    for  $j \leftarrow 1$  to  $c_b$ 
      for  $i \leftarrow 1$  to  $n$ 
        for  $l \leftarrow 1$  to  $c_0$ 
          if  $f_j^b(i) = f_l^0(i)$ 
            then  $M_j^b[i, l] \leftarrow 0$ 
          else  $M_j^b[i, l] \leftarrow 1$ 
  output ( $\{M_1^0, \dots, M_{c_0}^0\}, \{M_1^1, \dots, M_{c_1}^1\}$ )

```

Figure 1 A randomness preserving transformation from a BSS to a VCS

The proof of the following theorem, which has been omitted due to space constraints, can be found in the journal version of the present paper.

Theorem 3 *Let $C_0 = \{f_1, \dots, f_{c_0}\}$ and $C_1 = \{f_1, \dots, f_{c_1}\}$ realize a BSS for a strong access structure Γ on the set of n participants $P = \{1, \dots, n\}$. The algorithm described in Figure 1 generates a VCS on P for Γ with pixel expansion equal to $|C_0| = c_0$, contrast equal to one, and having the same randomness as the original BSS.*

Notice that by replacing each matrix M in the VCS of Theorem 3 with the matrix obtained by concatenating h copies of M , we obtain a VCS with contrast h and pixel expansion $h \cdot |C_0|$.

2.1. LOWER BOUNDS ON THE RANDOMNESS OF SECRET SHARING SCHEMES

Since visual cryptography schemes are a particular kind of binary secret sharing schemes, then any lower bound on the randomness of BSSs for a given access structure Γ is a lower bound on the randomness of any VCS for the

same access structure. Theorem 3 shows that the reverse implication holds as well, that is, any lower bound on the randomness of VCSs for the strong access structure Γ is also a lower bound on the randomness of any BSS for Γ . It follows that the techniques introduced in [8, 9] to derive lower bounds on the randomness of VCSs apply also to BSSs and consequently to secret sharing schemes for any set of secrets. In particular, the following lower bound [9] on the randomness of (k, n) -threshold VCS extends to any (k, n) -threshold secret sharing scheme:

$$(k - 1) \log(n - k + 2). \quad (2)$$

In [7] it has been proved that a (k, n) -threshold secret sharing scheme for a set of s secrets has randomness at least $(k - 1) \log s$. For set of secrets of size $s > n$, Shamir [14] has provided a scheme which achieves this bound. Then, one has that the following theorem holds.

Theorem 4 *For $n \geq k \geq 2$, the randomness of any (k, n) -threshold secret sharing scheme for a set of s secrets is at least $(k-1) \max\{\log s, \log(n-k+2)\}$.*

2.2. VCSs WITH PERFECT RECONSTRUCTION OF BLACK PIXELS

An important property of the VCSs obtained by applying the transformation of Figure 1 is that for any $X = \{i_1, i_2, \dots, i_p\} \in \Gamma_{\text{Qual}}$ and any $M \in \hat{\mathcal{C}}_1$, the “or” V of rows i_1, i_2, \dots, i_p consists of an all-one vector. VCSs with this property generate high quality images since they allow a perfect reconstruction of black pixels (see [6] for bounds on the pixel expansion of such VCSs). Given any VCS for the strong access structure Γ , we can construct a VCS with perfect reconstruction of black pixels for the same access structure as follows. We construct the distribution function collections C_0 and C_1 corresponding to the given VCS. Then, we apply the transformation of Figure 1 to obtain a VCS for Γ with perfect reconstruction of black pixels. By replacing each matrix $M \in \hat{\mathcal{C}}_0 \cup \hat{\mathcal{C}}_1$ with the matrix obtained by concatenating h copies of M , we obtain two matrix collections realizing a VCS with contrast h and with perfect reconstruction of black pixels. Hence, one has that the following theorem holds.

Theorem 5 *Let $\hat{\mathcal{C}}_0$ and $\hat{\mathcal{C}}_1$ be two matrix collections realizing a VCS for the strong access structure Γ . Then, for any arbitrary $h \geq 1$, there exists a VCS for Γ with perfect reconstruction of black pixels, having pixel expansion equal to $h \cdot |\hat{\mathcal{C}}_0|$, contrast equal to h , and the same randomness as the original VCS.*

The following example illustrates the above theorem.

Example 6 *Let us consider the strong access structure Γ on the set of participants $\{1, 2, 3, 4\}$ with basis $\Gamma_0 = \{\{1, 2\}, \{1, 3\}, \{2, 3\}, \{2, 4\}, \{3, 4\}\}$. The following matrix collections realize a VCS for Γ .*

$$\hat{c}_0 = \left\{ \begin{bmatrix} 10000 \\ 10001 \\ 10001 \\ 00001 \end{bmatrix}, \begin{bmatrix} 01000 \\ 01001 \\ 01001 \\ 00001 \end{bmatrix}, \begin{bmatrix} 00100 \\ 00101 \\ 00101 \\ 00001 \end{bmatrix}, \begin{bmatrix} 10000 \\ 10010 \\ 10010 \\ 00010 \end{bmatrix}, \begin{bmatrix} 01000 \\ 01010 \\ 01010 \\ 00010 \end{bmatrix}, \begin{bmatrix} 00100 \\ 00110 \\ 00110 \\ 00010 \end{bmatrix} \right\}$$

$$\hat{c}_1 = \left\{ \begin{bmatrix} 10000 \\ 01001 \\ 00101 \\ 00010 \end{bmatrix}, \begin{bmatrix} 01000 \\ 00101 \\ 10001 \\ 00010 \end{bmatrix}, \begin{bmatrix} 00100 \\ 10001 \\ 01001 \\ 00010 \end{bmatrix}, \begin{bmatrix} 10000 \\ 01010 \\ 00110 \\ 00001 \end{bmatrix}, \begin{bmatrix} 01000 \\ 00110 \\ 10010 \\ 00001 \end{bmatrix}, \begin{bmatrix} 00100 \\ 10010 \\ 01010 \\ 00001 \end{bmatrix} \right\}$$

The distribution function collections associated with this VCS are represented by the following two tables. For $i = 1,2,3,4$, the shares for participant i are denoted by a literal indexed with i . For a fixed index i , distinct literals indicates distinct shares.

	f_1^0	f_2^0	f_3^0	f_4^0	f_5^0	f_6^0
$T_0 =$	1	u_1	v_1	x_1	u_1	v_1
	2	u_2	v_2	x_2	y_2	z_2
	3	u_3	v_3	x_3	y_3	z_3
	4	u_4	u_4	u_4	v_4	v_4

	f_1^1	f_2^1	f_3^1	f_4^1	f_5^1	f_6^1
$T_1 =$	1	u_1	v_1	x_1	u_1	v_1
	2	v_2	x_2	u_2	z_2	w_2
	3	x_3	u_3	v_3	w_3	y_3
	4	v_4	v_4	v_4	u_4	u_4

Now we apply the randomness preserving transformation of Figure 1 to obtain a VCS for Γ with perfect reconstruction of the black pixels.

$$\hat{c}_0 = \left\{ \begin{bmatrix} 011011 \\ 011111 \\ 011111 \\ 000111 \end{bmatrix}, \begin{bmatrix} 101101 \\ 101111 \\ 101111 \\ 000111 \end{bmatrix}, \begin{bmatrix} 110110 \\ 110111 \\ 110111 \\ 000111 \end{bmatrix}, \begin{bmatrix} 011011 \\ 111011 \\ 111011 \\ 111000 \end{bmatrix}, \begin{bmatrix} 101101 \\ 111101 \\ 111101 \\ 111000 \end{bmatrix}, \begin{bmatrix} 110110 \\ 111110 \\ 111110 \\ 111000 \end{bmatrix} \right\}$$

$$\hat{c}_1 = \left\{ \begin{bmatrix} 011011 \\ 101111 \\ 110111 \\ 111000 \end{bmatrix}, \begin{bmatrix} 101101 \\ 110111 \\ 011111 \\ 111000 \end{bmatrix}, \begin{bmatrix} 110110 \\ 011111 \\ 101111 \\ 111000 \end{bmatrix}, \begin{bmatrix} 011011 \\ 111101 \\ 111110 \\ 000111 \end{bmatrix}, \begin{bmatrix} 101101 \\ 111110 \\ 111011 \\ 000111 \end{bmatrix}, \begin{bmatrix} 110110 \\ 111011 \\ 111101 \\ 000111 \end{bmatrix} \right\}$$

By concatenating h copies of each matrix in the above collections \hat{c}_0 and \hat{c}_1 we obtain a VCS with contrast h . △

3. A NEW UPPER BOUND ON THE RANDOMNESS OF (k, n) -THRESHOLD VCSs

In this section we provide a construction for (k, n) -threshold VCSs which improves on the randomness of all previously known VCSs and is very close to lower bound (2). The idea of the construction consists of applying Theorem 3 to Shamir's (k, n) -threshold secret sharing scheme [14]. Shamir's scheme shares a secret s , uniformly chosen in $GF(2^r)$, among a set of $n < 2^r$ participants. To share a secret s , the dealer uniformly and independently chooses $k - 1$ elements a_1, a_2, \dots, a_{k-1} in $GF(2^r)$ and then constructs the polynomial $p(x) = s + a_1x + a_2x^2 + \dots + a_{k-1}x^{k-1}$. The share assigned to participant i is $p(i)$. It is easy to see that if at least k participants join together then they can interpolate the polynomial $p(x)$ and calculate the secret $s = f(0)$, whereas any set of less than k participants has no information on the secret. The dealer uses

$(k - 1)r$ random bits to choose the coefficients a_1, a_2, \dots, a_{k-1} . The collection of distribution functions associated to a secret $s \in GF(2^r)$ is $C_s = \{p(x) = s + a_1x + a_2x^2 + \dots + a_{k-1}x^{k-1} : a_i \in GF(2^r), i = 1, \dots, k - 1\}$.

Given a Shamir's secret sharing scheme to share a secret $s \in GF(2^r)$ among a set of n participants, with $n < 2^r$, we can obtain a (k, n) -threshold BSS Σ as follows. Below, we will assume w.l.o.g. that the binary secret be chosen in $\{0, 1\}$. We assume that all secrets in $GF(2^r) \setminus \{0, 1\}$ be chosen with probability 0 and that the secrets 0 and 1 occur with probability $\frac{1}{2}$ each. To share a secret $s \in \{0, 1\}$, the dealer uniformly chooses a polynomial $p(x)$ in $C_s = \{p(x) = s + a_1x + a_2x^2 + \dots + a_{k-1}x^{k-1} : a_i \in GF(2^r), i = 1, \dots, k - 1\}$ and for $i = 1, \dots, n$, distributes to participant i the share $p(i)$. By applying the randomness preserving transformation of Figure 1 to Σ we obtain a VCS with randomness $(k - 1)r$. We can increase the contrast of the resulting VCS by replacing each matrix with h concatenated copies of that matrix. Since it must be $2^r > n$, then r can be as small as $\lceil \log(n + 1) \rceil$. Hence, the following theorem holds.

Theorem 7 For any $n \geq k \geq 2$ and $h \geq 1$, there exists a (k, n) -threshold VCS with pixel expansion $h \cdot 2^{(k-1)\lceil \log(n+1) \rceil}$, contrast h , and randomness $(k - 1)\lceil \log(n + 1) \rceil$.

Table 1 summarizes some known upper bounds on the randomness of (k, n) -threshold VCSs. Notice that the bound of Theorem 7 greatly improves on all other bounds. Indeed, all other bounds, except that of Corollary 2 of [9] which holds only for constant values of the threshold k , are exponential in k . Moreover, the upper bound of Theorem 7 is very close to lower bound (2).

Naor <i>et al.</i> [13]	$n^k \log(2^{k-1}!)$
Ateniese <i>et al.</i> [1]	$\log((O(k(2e)^k) \log n)!)$
Thm. 6 [9]	$\binom{n}{k-1} - 1$
Thm. 9 [9]	$(k - 1) \binom{n}{k}$
Cor. 1 [9]	$O(k^2 e^k) \log n$
Cor. 2 [9]	$O(k^2 \log^* n \log n)$, for k constant
Thm. 7	$(k - 1)\lceil \log(n + 1) \rceil$

Table 1 Upper bounds on the randomness of (k, n) -threshold VCSs.

3.1. MINIMUM RANDOMNESS (k, k)-THRESHOLD VCSs

In this section we show how to obtain a minimum randomness (k, k)-threshold VCS using the following well known construction for minimum randomness (k, k)-threshold BSSs (see for example [16]). To share a secret $s \in \{0, 1\}$ the dealer randomly chooses $k - 1$ random bits b_1, \dots, b_{k-1} and computes $b_k = s \oplus b_1 \oplus \dots \oplus b_{k-1}$, where “ \oplus ” denotes the “xor” operator. For $i = 1, \dots, k$, the share for participant i is b_i . It is easy to see that if k participants join together then they can recover the secret s by calculating the “xor” of their shares, whereas less than k participants have no information on s . The randomness of this BSS is $k - 1$. Hence, by applying the randomness preserving construction we obtain a VCS with pixel expansion 2^{k-1} , contrast 1 and randomness $k - 1$. By concatenating h copies of each matrix in the resulting VCS we obtain a minimum randomness (k, k)-threshold VCS with pixel expansion $h \cdot 2^{k-1}$ and contrast h .

The following example shows a (3,3)-threshold VCS obtained by applying the above construction.

Example 8 A minimum randomness (3,3)-threshold VCS with contrast $h = 1$.

$$\hat{C}_0 = \left\{ \begin{bmatrix} 0011 \\ 0101 \\ 0110 \end{bmatrix}, \begin{bmatrix} 0011 \\ 1010 \\ 1001 \end{bmatrix}, \begin{bmatrix} 1100 \\ 0101 \\ 1001 \end{bmatrix}, \begin{bmatrix} 1100 \\ 1010 \\ 0110 \end{bmatrix} \right\},$$

$$\hat{C}_1 = \left\{ \begin{bmatrix} 0011 \\ 0101 \\ 1001 \end{bmatrix}, \begin{bmatrix} 0011 \\ 1010 \\ 0110 \end{bmatrix}, \begin{bmatrix} 1100 \\ 0101 \\ 0110 \end{bmatrix}, \begin{bmatrix} 1100 \\ 1010 \\ 1001 \end{bmatrix} \right\}.$$

△

It is interesting to notice that the minimum randomness (k, k)-threshold VCS obtained in this section is also obtainable by using the construction for minimum randomness (k, k)-threshold VCSs provided in [9]. We recall that in [9] it has been shown that any (k, k)-threshold VCS with contrast h has pixel expansion larger than or equal to $h \cdot 2^{k-1}$ and that, for any value of the contrast h , our construction is the only one providing a (k, k)-threshold VCS with both minimum randomness and pixel expansion $h \cdot 2^{k-1}$.

4. CONCLUSIONS

In this paper we have provided a technique to transform a BSS into a VCS having the same randomness, thus proving that BSSs and VCSs are equivalent with respect to the randomness. Another consequence of our result is that any lower bound on the randomness of VCSs applies also to secret sharing schemes

for any set of secrets. A nice property of the VCSs obtained by applying our randomness preserving transformation is that they allow a perfect reconstruction of black pixels.

Our randomness preserving transformation has also been used to obtain a construction for (k, n) -threshold VCSs whose randomness is significantly smaller than the randomness of all previously known (k, n) -threshold VCSs and is very close to the known lower bound. An interesting open problem would be to further reduce the gap between the lower bound and the upper bound on the randomness of these VCSs.

References

- [1] G. Ateniese, C. Blundo, A. De Santis, and D. R. Stinson, *Visual Cryptography for General Access Structures. Information and Computation*, **129-2**, 86–106 (1996).
- [2] G. Ateniese, C. Blundo, A. De Santis, and D. R. Stinson, *Constructions and Bounds for Visual Cryptography. Proc. 23rd International Colloquium on Automata, Languages and Programming*, LNCS **1099**, 416–428 (1996).
- [3] A. Beimel and B. Chor, *Universally Ideal Secret Sharing Schemes. IEEE Trans. on Info. Theory*. **40**(3), 786–794 (1994) (Extended abstract in CRYPTO '92).
- [4] J. Benaloh and J. Leichter, *Generalized Secret Sharing and Monotone Functions. Lecture Notes in Computer Science*, **403**, 27–35 (1990).
- [5] G. R. Blakley, *Safeguarding Cryptographic Keys. AFIPS Conference Proceedings*, **48**, 313–317 (1979).
- [6] C. Blundo and A. De Santis, *Visual Cryptography Schemes with Perfect Reconstruction of Black Pixels. Journal for Computers & Graphics*, Special Issue: “Data Security in Image Communication and Network”, **22-4**, 449–455 (1998).
- [7] C. Blundo, A. De Santis, and U. Vaccaro, *Randomness in Distribution Protocols. Information and Computation*, **131**, 111–139 (1996).
- [8] A. De Bonis and A. De Santis, *New Results on the Randomness of Visual Cryptography, Proc. of Workshop on Cryptography and Computational Number Theory, CCNT'99*, Birkhauser, 187–201.
- [9] A. De Bonis and A. De Santis, *Randomness in Visual Cryptography. Proc. 17th International Symposium on Theoretical Aspects of Computer Science, STACS 2000*, LNCS **1770**, 626–638 (2000).
- [10] R. Impagliazzo and D. Zuckerman, *How to Recycle Random Bits. Proc. 21st Annual ACM Symp. on Theory of Computing*, 248–255 (1989).

- [11] M. Ito, A. Saito, and T. Nishizeki, *Secret Sharing Scheme Realizing General Access Structure*. *Proc. IEEE Global Telecommunications Conf., Globecom 87*, 99–102 (1987).
- [12] K. M. Martin, *New Secret Sharing Schemes from Old*. *J. of Combin. Math. and Combin. Comput.*, **14**, 65–77 (1993).
- [13] M. Naor and A. Shamir, *Visual Cryptography*. *Advances in Cryptology – EUROCRYPT '94*, LNCS **950**, 1–12 (1995).
- [14] A. Shamir, *How to Share a Secret*. *Commun. of the ACM*, **22**, 612–613 (1979).
- [15] D. R. Stinson, *An Introduction to Visual Cryptography*. Presented at *Public Key Solutions '97*, Toronto, Canada, April 28–30 (1997). Available as <http://bibd.unl.edu/stinson/VKS-PKS.ps>.
- [16] D. R. Stinson, *Cryptography, Theory and Practice*, 1995, CRC Press, Inc., Boca Raton, Florida.
- [17] G. J. Simmons, W. Jackson, and K. Martin, *The Geometry of Shared Secret Schemes*. *Bulletin of the ICA*, **1**, 71–88 (1991).
- [18] E. R. Verheul and H. C. A. van Tilborg, *Constructions and Properties of k out of n Visual Secret Sharing Schemes*. *Designs, Codes, and Cryptography* **11-2**, 179–196 (1997).
- [19] D. Zuckerman, *Simulating BPP Using a General Weak Random Source*. *Proc. 32nd IEEE Symp. on Foundations of Comp. Science*, 79–89 (1991).

This Page Intentionally Left Blank

A Two-level Time-Stamping System

Alban Gabillon and Jungsoo Byun

SIS/Equipe Informatique. Université de Toulon et du Var. {gabillon, byun}@univ-tln.fr

Key words: Time-stamping protocols, Time-Stamping Authority, One Way Accumulator, Verification protocol, Digital Notary Services

Abstract: Time-Stamping is a cryptographic technique which allows us to prove that an electronic document existed at a certain point in time and that it has not been modified since then. Different time-stamping schemes have already been proposed. Most of them use the concept of trusted Time-Stamping Authority (TSA). A TSA is in charge of time-stamping documents and delivering a time-stamping certificate for each time-stamped document. The purpose of this paper is to propose a new time-stamping scheme using a *Local Time-stamping System (LTS)*. The main idea can be summarised as follows: digests of the documents to be time-stamped are sent to a Local Time-stamping System (LTS). The LTS accumulates the digests into a *round value* using a round-based protocol. The round value is then time-stamped by a trusted and official TSA. We show how this time-stamping scheme could be useful for an organisation such as a digital library or a company.

1. INTRODUCTION

Like traditional paper documents, electronic documents need to be dated. However, electronic documents are easy to alter and forge. A date appended to a document can easily be replaced or modified by anybody having access to the document via a word processor.

Time-Stamping is a cryptographic technique which allows us to certify that an electronic document existed at a certain point in time and that it has not been modified since then. The first time-stamping protocol was proposed in [HS91]. A survey of the existing time-stamping protocols and of their security can be found in [MQ97][Pal98][Qal99][BLLV98][BLS00]. Most of

the existing time-stamping schemes use the concept of trusted Time-Stamping Authority (TSA). A TSA offers *digital notary services* that is, a TSA is able to securely time-stamp an electronic document. For example, If Alice wants to time-stamp a document d then she must proceed as follows:

- Using a well known one-way¹ collision-free² hashing function h , Alice first computes a *fingerprnt* (also called a *digest*) of the original document d .
- Alice sends the digest to the TSA. Notice that for confidentiality reasons document d is not sent to the TSA.
- The TSA sends back Alice a *time-stamping certificate* T_d which has been constructed with a particular *time-stamping protocol*.
- Alice can later present d and T_d to a verifier who needs to know when d was time-stamped, and who needs to make sure that d has not been modified since then. For this, the verifier must use the *verification protocol* associated with the time-stamping protocol which was used.

In most countries digital time-stamps and digital signatures have not received a legal value yet. Consequently, companies selling digital notary services and pretending to have the *authority* to time-stamp documents, have actually not been granted any official and legal authority. However, there is no doubt that in the near future, digitally signing and digitally time-stamping an electronic document will have the same legal force (and maybe more) than dating and signing a traditional paper document. At the same time, companies which will be officially and legally entitled to sell digital notary services will be clearly identified and organised. Most probably, they will integrate the internet Public Key Infrastructure (see [AT99] for an introduction to PKI and see [Aal00] for integrating TSA's to PKI). We, therefore, predict that the need for digitally signing and time-stamping electronic documents will then increase rapidly.

Now, let us consider an entity such as a company. The managers of the company may decide to time-stamp most of the electronic documents which are issued or received by the company. For this, they might ask each employee producing or receiving a given electronic document to contact a trusted and official TSA in order to have the document time-stamped. However, this scheme has some disadvantages:

- It is difficult to know which documents are time-stamped and which documents are not.
- It is expensive. The TSA will charge the company for each time-stamped document.

¹ one-way means that no portion of the original document can be reconstructed from the digest

² collision-free means that it is infeasible to find x and x' satisfying $h(x) = h(x')$

- Documents sent to the TSA by the employees might be personal documents which do not belong to the company.
- Documents cannot be time-stamped at a high rate.

The purpose of this paper is to propose a two-level time-stamping scheme using a *Local Time-stamping System* (LTS) and a trusted TSA. The main idea can be summarised as follows: if a particular employee has a document to time-stamp then this employee sends a digest of the document to the LTS. All the digests received by the LTS within a single round r are combined in order to produce the *round value* of the round r . This round value is then sent to a trusted TSA in order to be time-stamped.

In section 2 of this paper, we introduce the concept of LTS and we describe our 2-level time-stamping protocol as well as the corresponding verification protocol. In section 3, we present some possible solutions to manage time-stamped documents. In section 4, we discuss some security aspects. Section 5 concludes this paper.

2. LOCAL TIME-STAMPING SYSTEM

2.1 Principle

Behind the concept of LTS (Local Time-stamping System) there is a group of users. These users may be users working together in a particular organisation, for instance a company. As we have seen in the previous section, it is not realistic and efficient to ask an employee to contact a trusted TSA each time this employee has to time-stamp a document, especially if the company needs to have a high number of documents time-stamped everyday. The solution for this company is to install an LTS. In this section we describe the basic characteristics of such an LTS.

Our LTS uses a round-based protocol. A round has a certain duration which must be chosen by the administrator of the time-stamping system. It can be one second, one minute, one hour, one day. The smallest the round duration is, the better is the accuracy of the time-stamps. Documents which are time-stamped during the same round are all time-stamped with the same date and time by the TSA.

Let us assume that m documents d_1, d_2, \dots, d_m are to be time-stamped during round r . Their corresponding digests y_1, y_2, \dots, y_m are submitted to the LTS. These digests are combined in order to produce a single round value z which is called the *round value* of the round r (see next section 2.2). Once z has been calculated, it is sent to a distant trusted TSA. This TSA securely time-stamps z and returns a *time-stamping certificate for z* , $Cert_{TSA}(z)$. This

certificate consists of z , an announced date and time, and a secure time-stamp. The LTS can verify (with the verification protocol associated with the time-stamping protocol used by the TSA) that the announced date and time match the date and time included in the time-stamp. The LTS then produces a *time-stamping certificate for each document d_i* (with $1 \leq i \leq m$). This time-stamping certificate $Cert_{LTS}(d_i)$ consists of $y_i, z, Cert_{TSA}(z)$ and everything needed to prove that y_i participated in the construction of z .

Later, if a verifier wants to check the time-stamp of a given time-stamped document then he must be provided with the document and the corresponding time-stamping certificate. The verifier must hash the document and verify that the digest he obtains is equal to the digest included in the certificate. If they are equal then it proves that the document has not been modified since it was time-stamped. Then, the verifier checks whether the digest participated in the construction of the round value which is included in the certificate. Finally, the verifier checks the time-stamping certificate for the round value.

Some of the advantages of having a LTS are the followings:

- Documents can be produced at a high rate. There is no (theoretical) limitation on the number of documents which can be time-stamped within a particular round.
- The company is charged for a single certificate per round, without regard to the number of documents to be time-stamped during one round.
- The company has the possibility to control the time-stamping operations precisely. It can fix who has the right to time-stamp what. Auditing the time-stamping operations becomes easy.

Notice that we did not mention whether the documents which are time-stamped are digitally signed or not (see [RSA78] for a presentation of digital signatures). This is because there is absolutely no difference between time-stamping a signed document and time-stamping an unsigned document. If a document which is signed has to be time-stamped then the digest which is sent to the LTS is computed from the document with its signature appended to it.

Regarding this topic, let us mention that in [BHS92] it is shown that time-stamping a digitally signed document extends the life-time of the digital signature.

2.2 Protocol

Firstly we must say that the protocol used by the LTS is *not* a time-stamping protocol. It is a protocol which aims at securely constructing a single global time-stamping request from several local time-stamping requests. The global request is then submitted to the trusted TSA. The

protocol we have chosen for the LTS was first proposed in [BM94]. It uses a one-way accumulator.

A one-way accumulator is a one-way function owa which is *quasi-commutative*. This means that if one starts with an initial value x_0 and a set of values $y_1, y_2 \dots y_n$ then the accumulated hash

$$z = owa(owa(\dots owa(owa(x_0, y_1), y_2) \dots y_{n-1}), y_n)$$

would be unchanged if the order of the y_i were permuted. Consequently,

$$\text{if } z_i = owa(owa(\dots owa(owa(x_0, y_1), y_2) \dots y_{i-1}), y_{i+1}) \dots y_{n-1}), y_n) \text{ then } z = owa(z_i, y_i).$$

Proving that y_i participated in the construction of z means verifying that $z = owa(z_i, y_i)$.

One can refer to [BM94] in order to know more about accumulators.

Knowing this, we can now present the protocol of our LTS. The protocol uses the modular exponentiation as a one-way accumulator:

Let $y_1, y_2 \dots y_m$ be the digests of the documents $d_1, d_2, \dots d_m$ to be time-stamped during round r .

$y_i = h(d_i)$ with h being a well known one-way collision-free hashing function. See [MOS97] for a presentation of such functions.

Let x_0 be an initial value. Let $n = pq$ with p and q being two safe primes (see [BM94] for the definition of a safe prime).

Let z be the round value of the round. z is computed as follows:

$$z = x_0^{\prod_{i=1}^m y_i} \bmod n = ((\Lambda ((x_0^{y_1} \bmod n)^{y_2} \bmod n) \Lambda)^{y_m}) \bmod n.$$

The round value z is sent to the TSA for being time-stamped. The time-stamping certificate $Cert_{TSA}(z)$ is returned by the TSA.

For each digest y_j the *partial* round value z_j is computed by the LTS.

$$z_j = x_0^{\prod_{i=1}^m y_i} \bmod n.$$

Finally, the time-stamping certificate $Cert_{LST}(d_j)$ for the document d_j is produced by the LTS.

$$Cert_{LTS}(d_j) = (y_j, z_j, z, Cert_{TSA}(z)).$$

Note that the time-stamp is produced by the TSA. Presenting the scheme used by the TSA is irrelevant in this paper. However, we can mention that the scheme used by the TSA may be the scheme defined in [BLLV98] and later refined in [BLS00]. Indeed this scheme seems to be the most reliable of all the existing time-stamping schemes.

Note also that in [BM94], one-way accumulators are presented as a solution for time-stamping documents. Therefore, the TSA could also use a one-way accumulator. However, it has been shown that there is no efficient construction of accumulators without trapdoor (see [San99]).

In our paper, since we do not use accumulators for time-stamping, this trapdoor problem is not of our concern. We only use an accumulator for securely computing the round value which is time-stamped by the TSA. Using a one-way accumulator has the advantage of providing us with a very simple verification protocol.

2.3 Verification protocol

A verifier who needs to check the time-stamp of document d_j must be provided with the pair $(d_j, Cert_{LTS}(d_j))$. The verifier must check whether

1. $h(d_j) \stackrel{?}{=} y_j$ (has the document been modified since it was time-stamped ?)

2. $z_j^{y_j} \bmod n \stackrel{?}{=} z$ (did the document participate to the construction of z ?)

3. Finally the verifier must check $Cert_{TSA}(z)$ with the verification protocol corresponding to the time-stamping protocol used by the TSA.

3. MANAGING TIME-STAMPED DOCUMENTS

In the previous section, we did not mention anything about the management of the time-stamping certificates and the time-stamped documents. In this section we suggest two basic solutions for managing time-stamped documents efficiently.

3.1 Centralised management

Centralised management can be used by entities like digital libraries, e-print or e-publishers. Here, documents to be time-stamped are also documents to be *published*.

Instead of sending the digests, the users send the documents to the LTS. The LTS computes the digests itself³, and securely stores the time-stamping certificates and the time-stamped documents. Notice that documents sent by the users may be encrypted for confidentiality reasons using Secure Socket Layer.

Figure 1 represents the design of a digital library. Users of the library can submit their documents for time-stamping and publishing. The time-stamping certificates are stored in a Time-stamps Directory whereas the documents themselves are published in a Documents Directory. Anybody downloading a published document may also ask for the corresponding time-stamping certificate in order to verify the authenticity of the document. Notice that, the LTS may return a *copy* of $Cert_{LTS}(d_j)$ to the user who submitted d_j . This copy serves as an acknowledgement and has only an informational purpose.

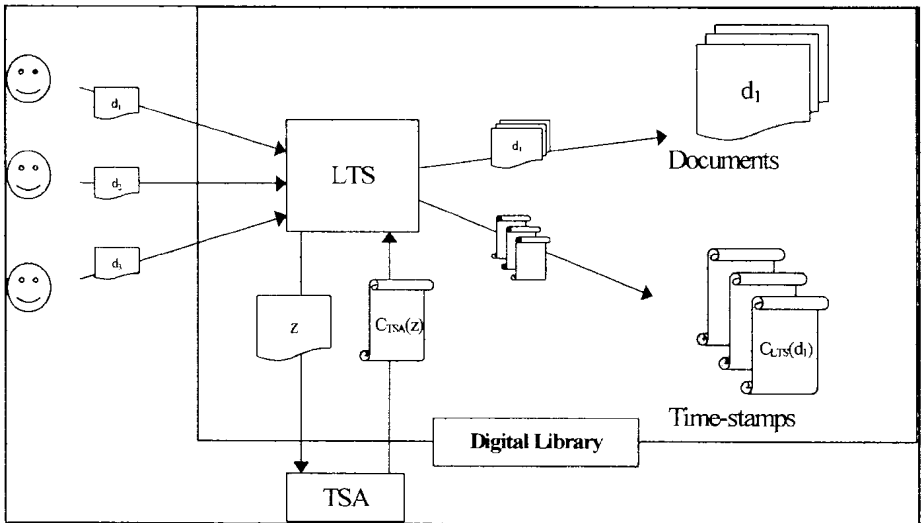


Figure 1. Centralised management of time-stamped documents

³ The LTS could directly use the document to compute the round value. However, since a digital document is big, the computation would be too slow. Therefore, it is better to hash each document before computing the round value.

3.2 Distributed management

Distributed management can be used by entities like companies. Here documents are time-stamped for internal management reasons. Users send the digests of the documents to the LTS. The LTS, in return, provides them with the time-stamping certificates. The documents and their corresponding time-stamping certificates are locally managed by each user.

Figure 2 represents the design of a company where each user actually acts on behalf of a particular department of the company. Time-stamped documents and their corresponding certificates are locally retained and managed at the departmental level.

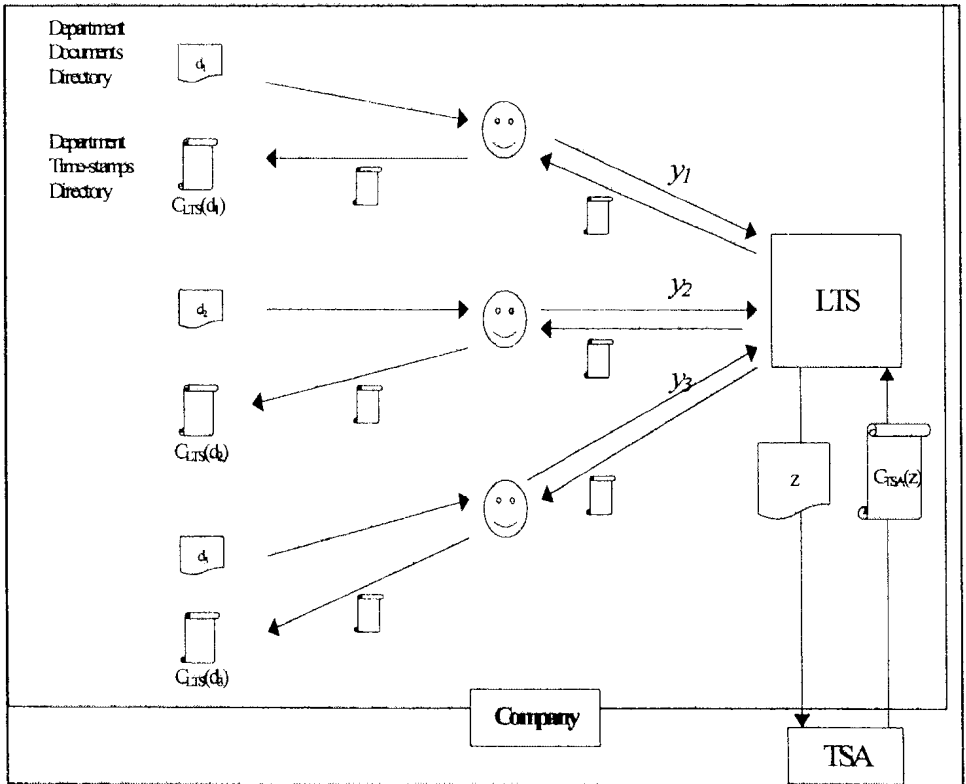


Figure 2. Distributed management of time-stamped documents

4. SECURITY

4.1 Security of the LTS

In the protocol used by the LTS, a round value is only used to securely accumulate several document digests in order to,

- centralise the time-stamping requests,
- pay for only one certificate,
- enable time-stamping at a high rate.

Since the LTS does not time-stamp the documents, it cannot backdate a document. Therefore, the only concern of the participants to the LTS should be to verify that the LTS cannot discard their requests or delay the execution of their requests. Notice that this potential problem, which is called a denial of service, is not specific to our LTS. All existing time-stamping systems cannot be trusted regarding this aspect.

In [Qal99], Quisquater et al. have designed a time-stamping system using an accumulator. However, their aim was to build a trusted system for a TSA. In order to make their system trusted, Quisquater et al. have proposed to link and accumulate the successive round values into a “big round value” which is regularly published into a widely witnessed media (like a daily newspaper). The verification protocol of [Qal99] includes steps 1 and 2 of our verification protocol but it also includes one more step: the verifier of a time-stamp must be provided with the round values belonging to the corresponding big round. By linking and accumulating these round values, the verifier can then reconstruct the corresponding trusted big round value. This technique ensures that, after a big round value has been published at a certain date, forging a time-stamp indicating an earlier date is impossible. With our LTS, we do not need to publish any round value. Indeed each round value is time-stamped by a TSA that we assume completely trusted and recognised as having a legal force.

The only real problem that we see might be the following: suppose the LTS sends two round values z_1 and z_2 successively to the TSA. Due to some network congestion round value z_2 reaches the TSA before round value z_1 . This may be possible especially if the round duration is short. Both round values are certified but z_2 receives a time-stamp indicating a date earlier than the date included in the time-stamp of z_1 ! This problem is actually a problem common to all time-stamping systems. It comes from the fact that the time which is taken into account for processing a request is not the time when the request is sent, but the time when the request is received.

4.2 User authentication

Basically, we can make the distinction between the following two types of users:

- Users who have the right to use the LTS.
- Users who have the right to consult the time-stamped documents and verify the time-stamps.

Users who have the right to consult the time-stamped documents and verify the time-stamps can be everybody or a very restricted number of people. This depends on whether the documents are public or whether they are private. In the case of a digital library (see section 3.1), we can assume that everybody has the right (for example through an anonymous access) to consult the documents and their corresponding time-stamps. Now, in the case of a company, it is clear that access to some of the documents will be restricted.

Users who have the right to use the LTS have to be registered by the LTS. The LTS must authenticate each user sending a time-stamping request.

5. CONCLUSION

In this paper, we presented a practical local time-stamping system which can serve as an intermediary between a group of users and an official and trusted TSA. We described the advantages of implementing a LTS in an organisation like a digital library or a company. However, further work remains to be done:

- We have to study how to implement a LTS. In particular we must define a policy for managing efficiently both the documents and the time-stamps directories.
- We need to define a security policy for managing the users authorisations. We must also propose a solution for implementing this policy (through an LDAP server for example).

Finally let us note that the Achilles' heel of our LTS is the TSA. Indeed commercial companies like Surety [Sur] selling digital notary services have not been granted any official authority to deliver time-stamping certificates. Companies forming the embryo of the internet PKI also lack such an official authority (see [ES00] for a discussion on this topic) and since, most probably, official TSA's will integrate the internet PKI, we must wait for a while before we can really have a TSA having a legal force at our disposal.

REFERENCES

- [Aal00] C. Adams, P. Cain, D. Pinkas, R. Zuccherato. *Internet X.509 Public Key Infrastructure Time Stamp Protocol*. Draft IETF. January 2001.
- [AT99] A. Arsenault and S. Turner. *Internet Public Key Infrastructure PKIX roadmap*. Draft 04, IETF, October 1999.
- [BHS92] D. Bayer S. Haber and W. Stornetta. *Improving the efficiency and reliability of digital time-stamping*. In Springer Verlag editor. Sequences' 91: Methods in Communication, Security and Computer Science, pages 329-334,1992.
- [BLB00] Ahto Buldas, Helger Lipmaa, Berry Schoenmakers. *Optimally Efficient Accountable Time-Stamping*. Public Key Cryptography '2000, volume 1751 of Lecture Notes in Computer Science, pages 293-305, Melbourne, Australia, 18--20 January 2000. Springer Verlag.
- [Bal98] Ahto Buldas, Peeter Laud, Helger Lipmaa, Jan Willemsen. *Time-Stamping with Binary Linking Schemes*. Advances in Cryptology --- CRYPTO '98, volume 1462 of Lecture Notes in Computer Science, pages 486-501. Springer-Verlag, 1998.
- [BM94] J. Benaloh and M. de Mare. *One-way accumulator: A decentralized alternative to digital signatures*. In Tor Hellesest, editor. Advances in Cryptology. Proceedings of Eurocrypt'93, number 765, pages 274-285. Springer-Verlag, 1994.
- [ES00] C. Ellison and B. Schneier. *Ten Risks of PKI: What you are not being told about PKI*. Computer Security Journal. Vol XVI, Number 1,2000.
- [HS91] S. Haber and W. Stornetta. *How to timestamp a digital document*. Journal of Cryptology, vol 3(2), pages 99-112,1991.
- [MOS97] A. Menezes, P. Oorschot and S. Vanstone. *Handbook of Applied Cryptography*, CRC Press, 1997.
- [MQ97] H. Massias, J.J. Quisquater. *Time and Cryptography*. Technical report, TIMESEC project, 1997.
- [Pal98] B. Preneel, B. Van Rompay, J.J. Quisquater, H. Massias, J. Serret Avila. *Design of a timestamping system*. Technical report, TIMESEC project, 1998.
- [Qal99] J.J. Quisquater, H. Massias, J. Serret Avila, B. Preneel, B. Van Rompay. *Specification and Implementation of a timestamping system*. Technical report, TIMESEC project, 1999.
- [RSA78] R. Rivest, A. Shamir, L. Adleman. *A method for obtaining digital signatures and public-key cryptosystems*. Communications of the ACM, 21, pages 120-126. 1978
- [San99] Tomas Sander. *Efficient Accumulators without Trapdoor*. In the second International Conference on Information and Communication Security, Sydney, Australia, 9-11 November 1999.
- [Sur] Surety Technology. <http://www.surety.com>

This Page Intentionally Left Blank

Security Analysis of the Cliques Protocols Suites: First Results

O. Pereira, J-J. Quisquater

UCL Crypto Group

Place du Levant, 3 - B-1348 Louvain-la-Neuve, Belgium.

{pereira, quisquater}@dice.ucl.ac.be

Key words: Group Protocols, Diffie-Hellman, Systematic Analysis, Cliques Protocols.

Abstract: The Cliques protocols are extensions of the Diffie-Hellman key exchange protocol to a group setting. In this paper, we are analysing the A-GDH.2 suite that is intended to allow a group to share an authenticated key and to perform dynamic changes in the group constitution (adding and deleting members, ...). We are proposing an original method to analyze these protocols and are presenting a number of unpublished flaws with respect to each of the main security properties claimed in protocol definitions (key authentication, perfect forward secrecy, resistance to known-keys attacks). Most of these flaws arise from the fact that using a group setting does not allow to reason about security properties in the same way as when only two (or three) parties are concerned.

1. INTRODUCTION

Within the scope of the CLIQUES project, five suites of group key distribution protocols have been developed. In this paper, we are studying the A-GDH.2 protocol suite [AST00]. The main A-GDH.2 protocol (that will be referenced as the A-GDH.2 protocol in the rest of this paper) allows a group of users to agree on a contributively generated key. The other

protocols of the suite permit the addition of new members in the group (A-GDH.2-MA), the removal of a member, the fusion of two groups, etc.

The analysis of these protocols raises a number of several problems that have not (or not much) been studied in the literature: taking into account low-level arithmetic properties, variable number of participants in the protocols, re-use of values in several protocols, . . . Furthermore, the intended security properties are not simple transpositions of those studied in the context of two parties protocols.

In this paper, we are proposing a simple model that we will use to reason about the A-GDH.2 protocol suite. The analysis we will perform with this model will lead us to the pinpointing of several attacks against these protocols. These attacks are typically performed by the intruder using the computations performed by honest users to obtain some secrets at the cost of the exclusion of a member from the group (which is computing a corrupted key or not receiving some messages). This exclusion, that would be very problematic in the case of two-parties protocols, has many chances to remain unnoticed by the other members of the group, particularly when the group size increases. It can also be interpreted as a network problem or as a temporary absence, which will not prevent the other members to use the key they computed.

This paper is organized as follows. First we will briefly define the A-GDH.2 protocols. Then we will explain the main particularities they present with respect to the usually analysed protocols and propose a model that we will use to perform our analysis. This analysis will constitute the last part of the text.

2. THE A-GDH.2 PROTOCOL SUITE

All protocols proposed within the scope of the CLIQUES project are based on the difficulty of a single problem: the Diffie-Hellman decision (DDH) problem (i.e. given a large integer p and knowing $\alpha^a \bmod p$ and $\alpha^b \bmod p$, it is difficult to compute $\alpha^{ab} \bmod p$). All arithmetic throughout this paper will be performed in a cyclic group G of prime order q which is a subgroup of Z_p for a prime p such that $p = kq + 1$ for small $k \in \mathbb{N}$ (e.g. $k = 2$). We assume that p , q and α are public and known by all users, and that every user M_i shares (or is able to share) with each M_j a distinct secret K_{ij} . For example, we can set $K_{ij} = F(\alpha^{x_i y_j} \bmod p)$ where x_i is a secret long-term exponent selected by every M_i and $\alpha^{x_i} \bmod p$ is the corresponding long-term public key. We will now describe the two protocols studied in this paper: the Key Generation and the Member Adding protocols (the other protocols of the suite are not described in detail in the literature).

2.1 The A-GDH.2 Protocol

Let $M = \{M_1, \dots, M_n\}$ be a set of users wishing to share a key S_n . The A-GDH.2 protocol executes in n rounds. In the first stage ($n - 1$ rounds), contributions are collected from individual group members and then, in the second stage (n -th round), the group keying material is broadcast. The actual protocol is as follows:

Initialization:

Let p be a prime integer and q a prime divisor of $p-1$. Let G be the unique cyclic subgroup of Z_p of order q , and let α be a generator of G .

Round i ($0 < i < n$):

1. M_i selects $r_i \in Z_q^*$
2. $M_i \rightarrow M_{i+1}: \{ \alpha^{r_j} \mid j \in [1, i] \}, \alpha^{r_1 \dots r_i}$

Round n :

1. M_n selects $r_n \in Z_q^*$
2. $M_n \rightarrow \text{All } M_i: \{ \alpha^{r_i} \mid i \in [1, n] \}$

Upon receipt of the above, every M_i computes the group key as:

$$\alpha^{\left(\frac{r_1 \dots r_n}{r_i} \cdot K_{in} \right) K_{in}^{-1} \cdot r_i} = \alpha^{r_1 \dots r_n} = S_n.$$

The main security properties that this protocol is intended to provide are the following:

- Implicit Key Authentication: each $M_i \in M$ is assured that no party $M_a \notin M$ can learn the key $S_n(M_i)$ (i.e. M_i 's view of the key) unless helped by a dishonest $M_j \in M$.
- Perfect Forward Secrecy: the compromise of long-term key(s) cannot result in the one of past session keys.
- Resistance to Known-Keys Attacks: the compromise of a session key cannot result in a passive adversary to compromise keys of other sessions, nor in an active adversary to impersonate one of protocol's parties.

All these properties have to be fulfilled in the presence of an active adversary who can insert, delay or delete messages.

2.2 The A-GDH.2-MA Protocol

Let $M = \{M_1, \dots, M_n\}$ be a set of users sharing a key S_n and assume that M_{n+1} is wishing to join the group. The A-GDH.2-MA protocol executes in 2 rounds: in the first one, M_n sends to M_{n+1} a message computed from the one

he broadcast in the last round of the A-DGH.2 protocol and from the old key while in the second round, M_{n+1} broadcast the new keying material to the group. The actual protocol is as follows:

Round 1:

1. M_n selects $\hat{r}_n \in \mathbb{Z}_q^*$
2. $M_n \rightarrow M_{n+1}: \{ \alpha^{\frac{\hat{r}_n^{r_1 \dots r_n} K_{in}}{r_i}} \mid i \in [1, n] \}, \alpha^{\hat{r}_n^{r_1 \dots r_n}}$

Round 2:

1. M_{n+1} selects $r_{n+1} \in \mathbb{Z}_q^*$
2. $M_{n+1} \rightarrow \text{All } M_i: \{ \alpha^{\frac{\hat{r}_n^{r_1 \dots r_{n+1}} K_{in} K_{in+1}}{r_i}} \mid i \in [1, n+1] \}$

Upon receipt of the above, every M_i (M_{n+1} included) computes the new key as:

$$\alpha^{(\frac{\hat{r}_n^{r_1 \dots r_{n+1}} K_{in} K_{in+1}}{r_i}) \cdot K_{in}^{-1} \cdot K_{in+1}^{-1} \cdot r_i} = \alpha^{\hat{r}_n^{r_1 \dots r_{n+1}}} = S_{n+1}.$$

The security properties described for the A-GDH.2 protocol are intended to be preserved after the execution of the A-GDH2.MA protocol.

3. A MODEL FOR THE ANALYSIS OF THE CLIQUE PROTOCOLS

A number of methods were developed during the last few years for the analysis of security protocols. Many of them are based on state-space exploration: they usually proceed by defining an arbitrarily bounded system and explore it hoping that if there is an error in the protocol, it can be described by a behavior included in the considered state-space ([MCJ97], [Low98], [DFG99], ...). However, several tools allow to obtain proofs for unbounded systems at the cost of the interactive proof of several lemmas [Mea96] or of the risk of receiving no answer for some protocols [Son99]. Other approaches are based on the use of logics ([SvO94], [Pau98], ...). They allow to obtain proofs for arbitrary size systems, but they often require error-prone formalization steps and does not provide the same support in pinpointing problems as the direct generation of counterexamples. Recently, “manual” approaches were presented, allowing to obtain fine-grained proofs for systems of any dimension, and even to analyze the interactions between protocols that can be executed concurrently (see [THG99a] for example).

In order to make such proofs feasible, several simplifying assumptions are typically stated: a very limited set of cryptographic primitives is

considered (typically public-key and symmetric-key encryption), and these primitives are usually idealized in such a way that they act as black-boxes (ignoring low-level properties such as the multiplicative structure of RSA or the characteristics of the chaining method used in symmetric-key encryption for example).

The use of state-space exploration techniques in the study of group-protocols seems very difficult due to their very essence : the number of participants in an honest session of the protocol is basically unbounded, what will intuitively result in dramatic state-space explosion problems. As we know, the only successful analysis of group protocols have been performed by theorem-proving approaches ([Pau97], [BS97]) which allow inductive reasoning. However we recently learned that C. Meadows was performing (independently of us) the analysis of the A-GDH.2 protocol, adapting her NRL Protocol analyser by extending the power and scope of its theorem-proving capabilities [Mea00].

Beyond the problem of the unbounded number of participants in the protocols, the modelling of the A-GDH.2 protocols suite requires the capturing of several low-level arithmetic properties: exponentiation, commutativity, associativity, that are out of the scope of most of the works encountered in the literature. Furthermore, the A-GDH.2 key generation protocol is not intended to be used alone: there are several other protocols in the suite (member addition, ...) that use values computed during the key generation protocol and can interfere with its security properties.

All these characteristics led us to try to adapt ideas presented in the context of the strand space approach ([THG99b], ...) in order to be able to reason about protocols based on the Diffie-Hellman Decision problem. In the following paragraphs, we will first introduce the modeling of the messages that we are using, then we will describe the intruder capabilities and, finally, we will show how the intended security properties can be verified and apply our method for the analysis of the A-GDH.2 protocols.

3.1 Messages and Intruder's Knowledge

The messages sent in the protocols proposed within the scope of the CLIQUES project are constituted by the concatenation of elements of a group G of prime order q that is a subgroup of Z_p (p and q being large prime integers). A particular element, that we will denote α , is a generator of G and is shared by all users of the network (as well as the knowledge of the characteristics of the group G). All exchanged elements of G are expressed as powers of $\alpha \pmod{p}$. It can then be checked that the participants have to manipulate three types of elements:

- Random Numbers (r_i)

- Long-term Keys (K_{ij})
- Elements of G expressed as α raised at the power of a product of random numbers and long-term keys. We will denote the set of all these product as P (i.e. $P = \left\{ \prod r_i^{e_i} \prod K_{j_l}^{e_{j_l}} \mid e_i, e_{j_l} \in \mathbb{Z} \right\}$). The only sent elements are the ones of this type.

The behaviour of the honest participants is quite simple: they receive elements of G , exponentiate them with random numbers and/or long-term key (possibly inverted), and send them to other participants. The group-key is obtained in the same manner, except that the result of the computations is not sent but kept confidential.

It can be noticed that when a participant receives an element of G , he has to accept it without being able to check anything concerning its constitution or origin. Furthermore, in the key-generation protocol described above, the completion of a protocol's session does not implies for any M_i the presence on the network of an other expected group member: the expected implicit key-authentication property says that the key computed by M_i at the reception of the broadcast of the n -th round of the protocol (key that we will write $S_n(M_i)$) can be known only by the participants to whom this message was broadcast by M_n (if M_n actually sent this message).

The goal of an intruder is hence to possess a pair of elements of G related between them in such a way that the second is equal to the result of the key-computation operations of a honest M_i applied to the first element of the pair, and there are n secret pairs corresponding to an execution of the protocol between n parties.

As we said above, the key-computation operation is always a sequence of exponentiations of a received element of G by some previously generated random numbers or keys. In a scriptural view, these operations amount to multiply an element of P by another (secret) element of P and to keep the result confidential. We can then define a set R as the set of the ratios between elements of P , and the goal of the intruder will be to obtain some secret value of R .

More precisely, our model will deal with two sets of elements:

- The set E containing the random numbers r_i and the long-term keys K_{ij}
- The set R of the ratios between elements of P . This set is defined as follows: given the set E and an injective function f from E to R , (R, \cdot) is the commutative group of which the elements of $Im(f)$ the image of E in R trough the f -function) are the generators. In order to simplify the notations, we will use the same letter to denote $e \in E$ and $f(e) \in R$.

Example: The pair $(\alpha^x, \alpha^{x r_1 K_{1n}^{-1}})$ will be represented by the element $r_1 K_{1n}^{-1} \in R$.

The use of such a construction implies several hypothesis concerning the elements of G . We have actually to assume that any element of G can only

be computed in one way (excepted the permutations in the order of the exponentiation of α and the possibility of exponentiation by an element of E and by its invert successively). In particular, we assume that $\alpha^{x+y} \neq \alpha^z$ (with $x, y, z \in P$) and, more generally, that a secret cannot be computed by combining elements of G (but only elements of E with elements of G). These hypotheses seem quite plausible given that we work within a large group and that the DDH problem is hard.

It can also be noticed that the use of the R -set implies another restriction due to its very structure: it does not allow to capture the relation between more than two elements of G . Once again, it does not seem to be a problem if we notice that the relevant security properties always come down to the impossibility of finding two elements of G presenting between them a particular relation, so that the consideration of more complex relations cannot be of any help to prove the correctness of the protocol. It could be useful to use such extensions to discover more dangerous attacks that violate more than one security property, but we are more interested in proving correctness than in finding “optimal” attack sketches.

We are currently working on the development of a more rigorous framework to express these hypothesis and determine the measure in which they are idealization of the real capabilities of the intruder. We will now be looking at the ways that the intruder can use to manipulate our two sets of elements.

3.2 Intruder Capabilities

In [STW96], M. Steiner & al. showed that the problem of computing $\alpha^{x_1 x_2 \dots x_n}$ from the view of the set of all $\alpha^{i_1 i_2 \dots i_m}$ where $\{i_1, i_2, \dots, i_m\}$ is a proper subset of $\{x_1, x_2, \dots, x_n\}$ was equivalent to the DDH problem. This can be used to convince us that the combination of several elements of G sent during a session of the protocol cannot be of any use in order to compute a secret element (but we are not providing any proof of it at this time).

The only computation that can be useful for the intruder will then be the exponentiation of an element of G by a known element of E . If we note E_I and R_I the subsets of elements of E and R (respectively) that are known by the intruder, we can then transpose this remark as follows:

$$(1) \text{ If } e \in E_I \text{ and } r \in R_I \text{ then } r.e \in R_I \text{ and } r.e^{-1} \in R_I$$

There is another way for the intruder to obtain new elements of R : the use of the computations executed by the honest users. As we said above, the behavior of these users is quite simple: they receive elements of G and

exponentiate them with some values of E . We will call such operations *services*. More precisely, a service is a function $s: G \rightarrow G, \alpha^x \rightarrow \alpha^{p \cdot x}$ ($x, p \in P$), and we call S the set of the available services. Let us see how a service can be described in term of growth of R_I . If $r \in R_I$, then the intruder possesses two elements of G that can be written α^x and α^{rx} . If the intruder sends α^x to an honest user performing the service $s: s(\alpha^x) = \alpha^{p \cdot x}$, then he will learn the element $p^{-1} \cdot r \in R_I$. Conversely, if the intruder sends α^{rx} to the user that performs the same service, he will learn the element $p \cdot r \in R_I$. We can then write our second rule for the increasing of the R_I -set:

$$(2) \text{ If } s \in S : s(\alpha^x) = \alpha^{p \cdot x}, \text{ and } r \in R_I \text{ then } r \cdot p \in R_I \text{ or } r \cdot p^{-1} \in R_I$$

Nevertheless, we have to be careful in the use of this rule and impose some restrictions in its application due to the fact that the honest users provide several services in parallel and only once. This will be examined more in the detail in the next section where we will propose a method to determine if a ratio is secret or can be obtained by the intruder.

3.3 Proving Secrecy Properties

In the context of the Cliques protocols, the most general message transformation provided by a user during a single round can be written as follows:

$$\alpha^{x_1} \alpha^{x_2} \dots \alpha^{x_n} \rightarrow \alpha^{x_1 y_{11}} \alpha^{x_1 y_{12}} \dots \alpha^{x_2 y_{21}} \alpha^{x_2 y_{22}} \dots \alpha^{x_n y_{n1}} \dots \alpha^{x_n y_{nn}}$$

This view can be used to express the rules limiting the composition of services in the derivation of the set R_I :

- The rule (2) can be used at most once for each service. Furthermore, it can only be used on an element of R_I that has been obtained previously.
- If two services $s_1: s_1(\alpha^x) = \alpha^{p_1 \cdot x}$ and $s_2: s_2(\alpha^x) = \alpha^{p_2 \cdot x}$ are performed during the same round and take distinct inputs (i.e. are applied to distinct α^{x_i}), then they can be used on a single element $r \in R_I$ to produce the following elements: $r \cdot p_1, r \cdot p_2, r \cdot p_1^{-1}, r \cdot p_2^{-1}, r \cdot p_1^{-1} \cdot p_2, r \cdot p_1 \cdot p_2^{-1}$ (but not $r \cdot p_1 \cdot p_2$ nor $r \cdot p_1^{-1} \cdot p_2^{-1}$)
- If two services $s_1: s_1(\alpha^x) = \alpha^{p_1 \cdot x}$ and $s_2: s_2(\alpha^x) = \alpha^{p_2 \cdot x}$ are performed during the same round and take the same inputs (i.e. are applied to the same α^{x_i}), then they can be used to produce the following elements: $p_1^{-1} \cdot p_2$ or $p_1 \cdot p_2^{-1}$. It can be noticed that these elements are independent from any previously known element of R_I .

From these considerations, we can suggest a general scheme to obtain the proof of the secrecy of a particular $r \in R$.

1. Expression of the available services (S -set), of the atomic elements and ratios initially known by the intruder (E_I and R_I), and of the secret ratios (let R_S be this set).
2. Suppression of all elements corresponding to those of E_I from the expression of S , R_I , and R_S . This operation simplifies the problem and does not change its solutions since:
 - If $e \in E_I$, every operation that uses the service $s: s(\alpha^x) \rightarrow \alpha^{xe^i y}$ ($i \in \mathbb{Z}$) can be performed by using a service $s': s'(\alpha^x) \rightarrow \alpha^{xy}$ and by suitably applying the (1)-rule.
 - If $e \in E_I$, and $r.e^a \in R_I$ then $r \in R_I$ (anew by applying rule (1))
 - If $e \in E_I$ and $r.e^a \in R_S$ then the knowledge of r implies the one of $r.e^a$ (for the same reason)

Example: if $K_{1n} \in E_I$ then the service $s: s(\alpha^x) \rightarrow \alpha^{x r_n K_{1n}}$ is as useful as the service $s': s'(\alpha^x) \rightarrow \alpha^{x r_n}$
3. Writing of the linear system expressing the “balance” of the variables in the construction of the secret from the services. This system contains one variable per service and element of R_I , one equation per element in E , and the second term of each equation is the power of the corresponding element in the studied secret. This system expresses that the only way to compute the secret is to successively apply some services on a known ratio. If this system is inconsistent, then the intended confidentiality property is verified (in our model). If this is not the case, we have to check the restrictions on the use of services described above. If it is possible to find a solution of the system that meets all these constraints, then an attack on the protocol can be derived.

We will now see how this scheme can be applied for the analysis of the A-GDH.2 protocols suite.

4. ANALYSIS OF THE A-GDH.2 PROTOCOLS SUITE

In the first paragraphs, we will concentrate our study on the properties of the A-GDH.2 key generation protocol. Then, we will extend it by considering the concurrent use of the A-GDH.2-MA protocol.

4.1 Analysis of the A-GDH.2 Key Generation Protocol

As described above, the first step in our analysis will be the description of the protocol.

In the first round, the user M_1 provides $r_1 \in R_I$. From the second round to the $(n-1)$ -th round, the user M_i provides the service $s_i: s_i(\alpha^x) \rightarrow \alpha^{x r_i}$ several

times in parallel. For the simplicity, we will refer to the service $s_i(\alpha^x) \rightarrow \alpha^{x r_i}$ by the power it raises its input : $r_i (\in S)$. During the n -th round, M_n provides the $n-1$ services: $r_n K_{1n}, \dots, r_n K_{n-1n}$. The secrets are the following: $r_i K_{in}^{-1}$ for $M_i (1 \leq i < n)$ and r_n for M_n .

Having so described the protocol, we will start our analysis by studying the implicit key authentication property, then we will turn to the perfect forward secrecy property, and finally to the resistance to known keys attacks property.

4.1.1 Implicit Key Authentication

In the study of this property, we can assume that the E_I -set is empty. If we follow the analysis scheme proposed above, we have now to express the linear system describing the “balance” of the variables of E . We will first look at the secrecy of $r_1 K_{1n}^{-1}$. If we use the “ s ”-letter to denote the coefficient of the variable indicating how many times the service s has to be used to construct the secret, it can be written as follows:

$$\begin{aligned} r_1 &= 1, r_2 = 0, \dots, r_{n-1} = 0 && \text{(balance on } r_1, \dots, r_{n-1}) \\ r_n K_{1n} + r_n K_{2n} + \dots + r_n K_{n-1n} &= 0 && \text{(balance on } r_n) \\ r_n K_{1n} = -1, r_n K_{2n} = 0, \dots, r_n K_{n-1n} &= 0 && \text{(balance on } K_{1n}, \dots, K_{n-1n}) \end{aligned}$$

It can be observed that the summing of the $n-1$ last equations provides an inconsistency with the n -th equation. Hence, we can say that $r_1 K_{1n}^{-1}$ cannot be obtained by using the two enrichment rules we defined and that $S_n(M_1)$ is kept secret in our model as claimed in protocol’s definition. If we write this system in the case of multiple sessions of the protocol (for which I is excluded), it can be easily checked that this inconsistency is preserved. The transposition of this result for the $r_i K_{in}^{-1}$ -secrets is straightforward and if we transform the second members of these equations in order to prove the secrecy of r_n , we can easily obtain an inconsistency between the same equations. We can then say that the Implicit Key Authentication property is correct with respect to our model.

4.1.2 Perfect Forward Secrecy

In the study of this property, we will assume that E_I contains all long term keys K_{in} . If we apply the transformation suggested as second step of our proof-scheme, we can rewrite the set of services $S = \{r_i \mid i \in [2, n]\}$, $R_I = \{r_1\}$, and R_S as $\{r_i \mid (i \in [1, n])\}$. For each secret r_i , the resulting linear system has a trivial solution: $r_i = 1$. This solutions meets all restrictions described above, and we can then assume that the perfect forward secrecy is somehow suspicious.

A scenario corresponding to an attack against M_1 is as follows. The secret is $r_1 K_{1n}^{-1}$ and the value of interest is r_1 provided by M_1 in the first round. The intruder will therefore replace the element of G intended to M_1 in the broadcast of the n -th round by a in such a way that $S_n(M_1)$ will be computed as $\alpha^{r_1 K_{1n}^{-1}}$. The perfect forward secrecy property says that the compromising of long term keys cannot result in the one of session keys. But if K_{1n} is compromised, the intruder will be able to compute $\alpha^{r_1 K_{1n}^{-1}}$ (by exponentiating α^{r_1} provided during the first round). Hopefully, this problem does not seem very dangerous in the practice since $S_n(M_1)$ will be different of the keys computed by the other members of the group. The scenario will be similar for all the other M_i ($i < n$), and it can be noticed that all these attacks can be performed in parallel, which can be useful in some contexts. However, the attack against M_n will be somewhat different. His secret is r_n , and the useful services are $r_n K_{1n}, \dots, r_n K_{n-1n}$ (each can be used). These services are respectively applied to the $n-1$ first elements of the $(n-1)$ -th round, and the secret is computed from the last element of the same run. The intruder will then proceed by substituting one of the $n-1$ first elements of this round with the last element of the message. If we suppose that he substitutes the first element, M_n will compute $S_n(M_n) = \alpha^{r_1 \dots r_n}$ and broadcast $\alpha^{r_1 \dots r_n K_{1n}}, \alpha^{r_1 r_3 \dots r_n K_{2n}}, \dots, \alpha^{r_1 r_2 \dots r_{n-2} r_n K_{n-1n}}$. Hence M_1 will be computing a wrong key: $S_n(M_1) = \alpha^{r_1 \dots r_n}$ while all other members of the group will compute $S_n(M_i) = \alpha^{r_1 \dots r_n}$ ($1 < i \leq n$). Then, if K_{1n} is compromised, the intruder will be able to compute the key $S_n(M_n)$ that is shared by all group-members except one (that he can isolate from the rest of the network or that can have never been alive). This attack is represented for a group of four members in Fig. 1. It seems to us that this is a much more awkward scenario. The fact that this attack provides the key computed by group-members others than M_n corresponds to the fact that it exploits solutions of the type $r_i = 1, r_n K_{in} = -1, r_n K_{jn} = 1$ that are less trivial solutions of the system corresponding to the secret of M_i .

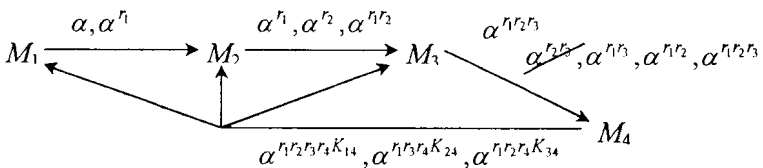


Figure 1. Attack against the Perfect Forward Secrecy Property

4.1.3 Resistance to known-keys attacks

This property expresses that the compromising of session keys does not allow a passive adversary to compromise keys of other sessions nor an active adversary to impersonate one of the protocol parties. The part of this

property concerning the passive adversary is studied in [AST00] and we will focus on the second part.

However the authors claim that the resistance to an active adversary is more dubious and suggest an attack that does not seem very useful in the practice. The application of our method to the verification of this property is as follows. We will assume two sessions of the protocol with the same participants, and the random numbers generated by M_i during the first and second sessions of the protocol will be r_i and r_i' respectively. Hence, we can write that:

$$\begin{aligned} S &= \{ r_2, \dots, r_{n-1}, r_n K_{1n}, \dots, r_n K_{n-1n}, r_2', \dots, r_{n-1}', r_n' K_{1n}, \dots, r_n' K_{n-1n} \} \\ E_j &= \emptyset \\ R_j &= \{ r_1, r_1', r_1 K_{1n}^{-1}, \dots, r_{n-1} K_{n-1n}^{-1}, r_n \} \\ R_S &= \{ r_1' K_{1n}^{-1}, \dots, r_{n-1}' K_{n-1n}^{-1}, r_n' \} \end{aligned}$$

If we write the linear system corresponding to $r_i' K_{in}^{-1}$ ($1 \leq i < n$), we can check that

$$r_i K_{in}^{-1} = 1, r_i = -1, r_i' = 1$$

and all other services unused is a solution. If $i = 1$, it is however impossible to find an attack scheme since all these values are in R_j and cannot be successfully assembled. Nevertheless, for all others values of i , the following attack is possible:

1. Let α^x be one of the terms of the input of the i -th round of the first run of the protocol. M_i will therefore send $\alpha^{x r_i}$.
2. The intruder replaces then the term $\alpha^{r_1 \dots r_{i-1} r_{i+1} \dots r_n K_{in}}$ with α^x . Hence $S_n(M_i)$ will be equal to $\alpha^{x r_i K_{in}^{-1}}$. Since we study known-keys attacks, we will assume that this value is compromised.
3. In the second run of the protocol, the intruder replaces one of the inputs of the i -th round with $\alpha^{x r_i K_{in}^{-1}}$. M_i will therefore send $\alpha^{x r_i K_{in}^{-1} r_i'}$.
4. In the broadcast of the second run of the protocol, the intruder finally replaces the term intended to M_i with $\alpha^{x r_i}$ (obtained in the first step of our scenario). Hence $S_n'(M_i)$ will be computed as $\alpha^{x r_i K_{in}^{-1} r_i'}$ that has been obtained during the third step of our scenario.

At the end of this scenario, the intruder will possess a key that M_i believes to be secret. However this key is unknown to the rest of the group and the compromised key used is a malformed key which reduces the scope of these attacks. However, if all malformed keys are available, the intruder can perform this attack simultaneously against almost all members of the group!

We can now turn to the secrecy of r_n . If we look at the linear system corresponding to this secret, we can find two types of solutions. The first is:

$$r_i = -1, r_i K_{in}^{-1} = 1, r_n' K_{in} = 1$$

From these solutions, we can obtain the scenarios corresponding to the attack proposed in [AST00]. The scope of these attacks is the same as the one we just described.

However, another type of solution can be found:

$$r_n = 1, r_n K_{in} = -1, r_n' K_{in} = 1$$

For $1 \leq i < n$, it is possible to apply the following scenario:

1. In the inputs of the last round of the first session of the protocol, the intruder replaces $\alpha^{r_1 \dots r_{i-1} r_{i+1} \dots r_{n-1}}$ with $\alpha^{r_1 \dots r_{i-1} r_{n-1}}$. Hence, all elements of the broadcast will be preserved except the one intended to M_i that will be equal to $\alpha^{r_1 \dots r_n K_{in}}$. $S_n(M_n)$ will hence be equal to $\alpha^{r_1 \dots r_n}$ and shared by all members of the group except M_i . In a context of known-key attacks, we will assume that this key is compromised.
2. In the inputs of the last round of the protocol second session, the intruder will substitute $\alpha^{r_1 \dots r_{n-1}}$ with $\alpha^{r_1 \dots r_n K_{in}}$ and $\alpha^{r_1 \dots r_{i-1} r_{i+1} \dots r_{n-1}}$ with $\alpha^{r_1 \dots r_n}$. Hence M_n will broadcast $\alpha^{r_1 \dots r_n K_{in} r_n'}$ and compute $S_n(M_n) = \alpha^{r_1 \dots r_n K_{in} r_n'}$.

This scenario is more dangerous since we assume the compromising of a key that has been shared (and normally used) by all members of the group except one. However, it allows to attack only M_n . Fig 2. represents this scenario for $i=1$ and a group of four members.

Consequently, the resistance to known-key attacks seems problematic in this protocol.

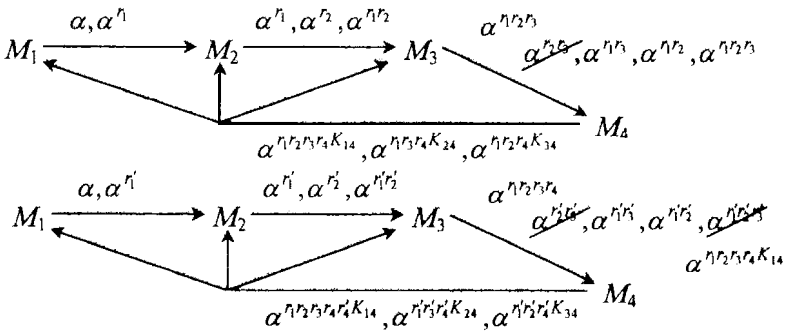


Figure 2. Attack against the Resistance to Known-Keys Property

4.2 Consideration of the Use of the A-GDH.2-MA Protocol

The key generation protocol (A-GDH.2) is not intended to be used alone: it is often useful to enable the addition or deletion of group members after the initial group creation and, in order to provide each of these services, we will use new protocols. As we said above, the aim of the A-GDH.2-MA protocol is the addition of a new member in the group. In this paragraph, we will extend our analysis of the A-GDH.2 protocol by taking into account the presence of the Member Adding protocol.

As a first step, we will study the Implicit Key Authentication property and consider two sessions of the protocols: in the first session, the A-GDH.2 protocol is executed by M_1, \dots, M_n ; while in the second session a member is added to this group. Following the same approach as above, we will first write the sets E_I, R_I, R_S and S that will be the union of those corresponding to each of the two protocols sessions:

$$S = \{ r_2, \dots, r_{n-1}, r_n K_{1n}, \dots, r_n K_{n-1n}, \quad (\text{A-GDH.2 Protocol}) \\ r_n \hat{r}_n, r_n \hat{r}_n K_{1n}, \dots, r_n \hat{r}_n K_{n-1n}, \hat{r}_n K_{nn}, \quad (\text{First round of A-GDH.2-MA}) \\ r_{n+1} K_{1n+1}, \dots, r_{n+1} K_{n+1n+1} \} \quad (\text{Last round of A-GDH.2-MA})$$

$$E_I = \emptyset$$

$$R_I = \{ r_1 \}$$

$$R_S = \{ r_1 K_{1n}^{-1}, \dots, r_{n-1} K_{n-1n}^{-1}, r_n, \quad (\text{A-GDH.2 Protocol})$$

$$r_1 K_{1n}^{-1} K_{1n+1}^{-1}, \dots, r_{n+1} K_{n+1n}^{-1} K_{n+1n+1}^{-1} \} \quad (\text{A-GDH.2-MA Protocol})$$

\bar{E}_I being empty, we can immediately study the linear system corresponding to the secrets. This system is a little larger than the previous but remains quite regular. If we solve it, we find that a number of secrets can be compromised: $r_i K_{in}^{-1}$ ($1 \leq i < n$) can be obtained by combining the services (or ratios in the case of r_1) $r_i, r_n \hat{r}_n$ and $r_n \hat{r}_n K_{in}$ ($1 \leq i < n$). The other secrets can not be compromised in this scheme. The corresponding scenario is as follows:

1. M_1, \dots, M_n execute the key-generation protocol, but I intercepts the broadcast of the n -th round.
2. I obtains that M_n starts the A-GDH.2-MA protocol with some other user of the network, and eavesdrop the first message.
3. I sends the parts corresponding to the users M_1, \dots, M_{n-1} faking the broadcast of the A-GDH.2 protocol.

When done, M_1, \dots, M_{n-1} will share with the intruder the key $\alpha^{r_1 r_2 \dots r_n \hat{r}_n}$ that has been sent by M_n as the last part of the first message of the A-GDH.2-MA protocol. The scheme corresponding to this attack in the case of the adding a fourth member to the group is described in Fig. 3. Hence the Implicit Key Authentication property seems to become problematic when we consider the possibility of the use of the A-GDH.2-MA protocol in parallel with the A-

GDH.2 protocol. This security property being compromised, it does not seem useful to continue our analysis for the other properties.

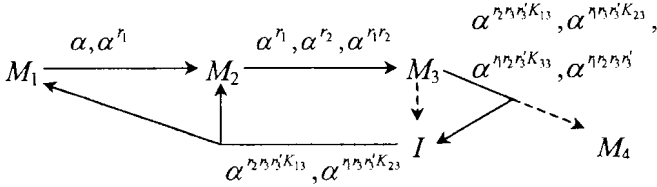


Figure 3. Attack against the Implicit Key Authentication Property

5. CONCLUSION

Throughout this paper, we presented the first steps of the development of a model for the analysis of the Cliques protocols. The reasoning in our model led us to pinpoint a number of unpublished flaws in the A-GDH.2 protocols suite, emphasizing the necessity to be able to reason systematically on security protocols, especially in contexts where active adversaries are to be considered. The scope of these flaws is summarized in the following Table.

Table 1. Summary of the Flaws

Protocols Considered	Property Analysed	Number of Members Flawed
A-GDH.2	Implicit Key Authentication	0
A-GDH.2	Perfect Forward Secrecy	$n-1$
A-GDH.2	Resistance to Known-Keys	1 (but parallel attacks possible)
A-GDH.2 and A-GDH.2-MA	Implicit Key Authentication	$n-1$

We are currently working on defining more precisely the attacks detectable (and those undetectable) with our model, on the incorporation of our “machinery” in more general models, and on the construction of fixes on the A-GDH.2 protocols that are secure from our model point of view.

Acknowledgements

The authors would like to thank O. Chevassut for his useful comments on the Cliques protocols.

References

- [AST00] G. Ateniese, M. Steiner and G. Tsudik. *New Multi-party Authentication Services and Key Agreement Protocols*. IEEE Journal on Selected Areas in Communication, April 2000.
- [BS97] J. Bryans, S. Schneider. CSP, PVS, and a Recursive Authentication Protocol. In DIMACS Workshop on Formal Verification of Security Protocols, 1997.
- [DFG99] A. Durante, R. Focardi, R. Gorrieri. *CVS: A Tool for the Analysis of Cryptographic Protocols*. In Proceedings of the 12-th IEEE Computer Security Foundations Workshop, pp. 203-212, 1999.
- [Low98] G. Lowe. *Casper: A Compiler for the Analysis of Security Protocols*. In Journal of Computer Security, Vol. 6, pp. 53-84, 1998.
- [Mea96] C. Meadows. *The NRL Protocol Analyzer : an Overview*. In Journal of Logic Programming, Vol. 26(2), pp. 113-131, 1996.
- [Mea00] C. Meadows. *Extending Formal Cryptographic Protocol Analysis Techniques for Group Protocols and Low-Level Cryptographic Primitives*. In Proceedings of the Workshop on Issues in the Theory of Security (WITS 2000), 2000.
- [MCJ97] W. Marrero, E. Clarke, S. Jha. *A Model Checker for Authentication protocols*. In Proceedings of the DIMACS workshop on design and formal verification of security protocols, 1997.
- [Pau97] L C Paulson. *Mechanised Proofs for a Recursive Authentication Protocol*. In Proceedings of the 10th Computer Security Foundations Workshop, pp. 84-95. IEEE Computer Society Press, 1997.
- [Pau98] L C Paulson. *The inductive approach to verifying cryptographic protocols*. In Journal of computer Security, Vol. 6, pp. 85-128, 1998.
- [Son99] D. Song. *Athena: A New Efficient Automatic Checker for Security Protocol Analysis*. In Proceedings of the IEEE Symposium on Research in Security and Privacy, 1999.
- [STW96] M. Steiner, G. Tsudik and M. Waidner. *Diffie-Hellman Key Distribution Extended to Group Communication*. In Proceedings of the 3rd ACM Conference on Computer and Communications Security, 1996.
- [SvO94] P. Syverson, P. van Oorschot. *On Unifying Some Cryptographic Protocols logics* ». In Proceedings of the IEEE Symposium on Research in Security and Privacy, pages 14-28, 1994.
- [THG99a] F. J. Thayer, J. H. Herzog, J. Guttman. *Strand Spaces: Proving Security Protocols Correct*. In Journal of Computer Security, 7(2/3): 191-230, 1999.
- [THG99b] F.J. Thayer, J. H. Herzog, J. Guttman. *Mixed Strand Spaces*. In Proceedings of the 12th Computer Security Foundations Workshop, pp. 83-89. IEEE Computer Society Press, 1999.

A SECURE WORKFLOW SYSTEM FOR DYNAMIC COLLABORATION

Joon S. Park, Myong H. Kang, and Judith N. Froscher

Center for High Assurance Computer Systems

US Naval Research Laboratory

{jpark, mkang, froscher} @ ltd.nrl.navy.mil

Abstract The emergence of the Internet has broken down geographic and organizational boundaries, providing a virtual common workplace regardless of the heterogeneity of participating organizations. Enterprise projects that used to be done autonomously now span multiple organizations. While an inter-organizational workflow, as one of several technologies supporting inter-organizational collaboration, provides an easy-to-use collaborative work environment for users, it also increases the complexity of security maintenance and brings about security problems that were not considered before. Unconventional collaborations among businesses and organizations are formed to advance common goals. In this paper, we address the security services to support inter-organizational collaborative enterprises, which may span multiple organizations, and describe how we develop a secure workflow system to satisfy the requirements by integrating with existing, wellknown technologies. Although we apply our ideas to particular technologies, such as workflows and RBAC, in this paper, we believe it is always possible to apply our approaches to other systems, which support many users from different organizations.

Keywords: Dynamic Collaboration, Information Security, Role-based Access Control (RBAC), Workflow

1. INTRODUCTION

In the days before the ubiquitous Internet and its use across all industries, collaborative projects were planned in accordance with geographic and organizational borders. The emergence of the Internet has broken down these boundaries, providing a virtual common workplace. Organizations can communicate with suppliers and partners, and with customers more efficiently and effectively. Enterprises that were autonomous now span multiple organizations, which may

join or leave an enterprise project dynamically while the project is still underway.

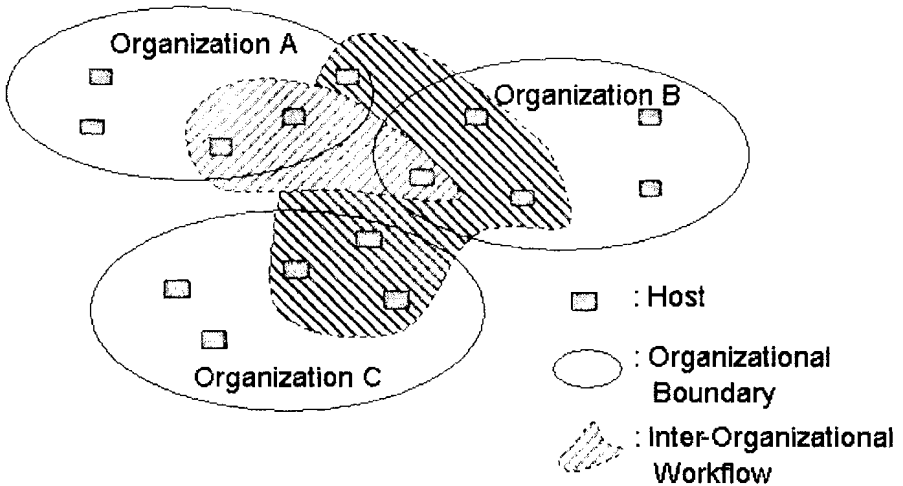


Figure 1 An Example of Inter-Organizational Workflows

One technology that can satisfy this service is inter-organizational workflow. We consider an inter-organizational workflow as a virtual enterprise in this paper. Figure 1 shows an example of inter-organizational workflows, which span multiple organizations to conduct their missions. In this example, one workflow spans two organizations while the other workflow spans three organizations. Once a workflow is designed, each task is conducted in a specific host (machine) in a specific organization. The hosts are connected via the Internet and may support multiple tasks for multiple enterprises (particularly, workflows in this paper). Individual users conduct their human tasks by connecting to a specific machine in a specific domain, while non-human tasks are executed automatically on demand under the workflow policy.

While an inter-organizational workflow supports an easy-to-use collaborative work environment for users, it also increases the complexity of security maintenance and causes new security problems that did not appear in autonomous enterprises. For example, how can we control efficiently and securely who is doing what and when? Unconventional collaborations among businesses and organizations are formed to advance common goals. These collaborations may quickly dissolve as individual objectives change (we call these dynamic collaborations in this paper). Threats now lie in these essential connections among participating organizations, which may be involved in multiple enter-

prises across other organizations. Therefore, there is the need for new types of security services for the common workplace, which provides a collaborative work environment.

For a secure workflow, especially, if dynamic collaboration is necessary, we need the following security services.

- Providing secure communication between components and users
- Separating security infrastructures between organizations and enterprises (workflows)
- Providing different privileges to different users
- Validating enterprise (workflow) design

In this paper, we describe why we need to satisfy the above requirements for a dynamic workflow system, and how we have implemented each in our system by integrating with existing security technologies. Although we describe our approaches within a workflow management system that we have developed, we believe that the technologies we introduce in this paper can be easily applied to other systems, which support many users, requiring security services between components and users.

The rest of this paper is organized as follows. In Section 2, we briefly introduce the existing technologies that we use for our implementation, including OrbWork, RBAC (Role-based Access Control), and SSL (Secure Socket Layer). Section 3 describes the system architecture of our secure workflow management system. In Section 4, we describe how we provide security services to our workflow system for dynamic collaboration. Finally, Section 5 summarizes our implementation and concludes this paper.

2. RELATED WORK

2.1. ORBWORK

Researchers at the University of Georgia developed a workflow enactment service, OrbWork [10] in 1998. OrbWork is a single-level distributed workflow engine that exploits CORBA (Common Object Request Broker Architecture), JAVA, and Web technologies. It does not have a central scheduler; rather it is distributed with a scheduler per task. Each scheduler only knows its predecessors and successors.

Basically, OrbWork consists of the following CORBA servers: task servers, worklist servers, and data servers. Figure 2 shows how the OrbWork components interact with each other. Each task server may contain more than one task. Each task has three parts: task scheduler, task manager and the underlying component. The worklist server maintains the lists of pending work for

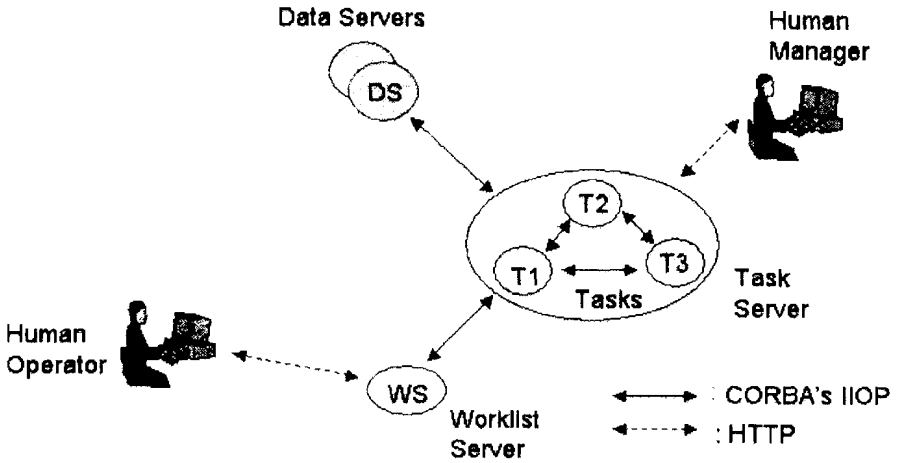


Figure 2 Communications Among the OrbWork Components

human tasks. Data servers act as repositories for data that need to be accessed by tasks. Since they are CORBA servers, they communicate with each other through CORBA's IIOP (Internet Inter-ORB Protocol).

The task and worklist servers are not only CORBA servers but also HTTP (HyperText Transfer Protocol) servers. When a human operator has to interact with the worklist server (e.g., human task), he can do so through HTTP. Also when a human workflow manager needs to intervene in task servers for some reason, he can do so through HTTP. Currently, the original OrbWork does not provide security services among its components and between its components and users.

2.2. ROLE-BASED ACCESS CONTROL

A large workflow system is usually designed to support many users. Some users need to be temporarily involved in the workflow. A user may need to have different access privileges based on his context, while other users may need to have the same privilege. If we use the conventional identity-based access control mechanism, it is very hard to determine and control which permissions should be authorized for what users, especially, in a large system. The direct mapping between users and permissions is transitory and brings very inefficient management. Therefore, we have decided to use Role-based Access Control (RBAC [18]) for our secure workflow management system for dynamic collaboration.

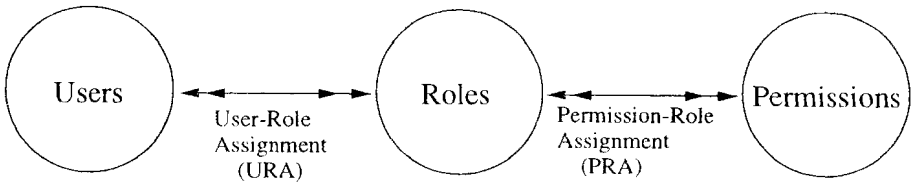


Figure 3 A Simplified RBAC Model

RBAC has rapidly emerged in the 1990s as a technology for managing and enforcing security in large-scale enterprise-wide systems. The basic notion of RBAC is that permissions are associated with roles, and users are assigned to appropriate roles thereby acquiring the roles' permissions. Figure 3 shows a simplified RBAC model. RBAC ensures that only authorized users are given access to certain data or resources.

In RBAC, a role is a semantic construct forming the basis of an access control policy. System administrators can create roles, grant permissions to those roles, and then assign users to the roles on the basis of their specific job responsibilities and policy.

RBAC separates the mapping between users and permissions through User-Role Assignment (URA) and Permission-Role Assignment (PRA). Usually, PRA is more stable (of course it can be changed if it is necessary) than URA, because job responsibilities in an organization do not change frequently while users' job functions change quite often. The system makes access control decisions based on the users roles instead of their identities. This provides an efficient access control mechanism to the system and resolves the scalability problem.

To implement the RBAC model on the Web, Park and Sandhu have identified two different approaches for obtaining a user's roles, especially, with respect to user-pull and server-pull architectures [13]. Basically, there are three components in both architectures: client, Web server, and role server. Clients connect to Web servers via HTTP using browsers. The role server is maintained by an administrator and assigns users to the roles in the domain. In the user-pull architecture, a user pulls his roles from the role server and then presents them to the Web servers. In the server-pull architecture, each Web server pulls the user's roles from the role server as needed and uses them for RBAC. Comprehensive descriptions and tradeoffs between the two different RBAC architectures are discussed in [16]. In this paper, we apply those approaches to build our secure workflow system, providing RBAC services in individual task servers (described in Section 2.1). Detailed technologies (such as authentication, role

transfer and protection, and verification) to support these architectures depend on the applications and environments.

2.3. SECURE SOCKET LAYER (SSL) PROTOCOL

The SSL protocol [21] was introduced with the Netscape Navigator browser in 1994, and rapidly became the predominant security protocol on the Web. Since the protocol operates at the transport layer, any program that uses TCP (Transmission Control Protocol) is ready to use SSL connections. The SSL protocol provides a secure means for establishing encrypted communication between Web servers and browsers. SSL also supports the authentication service between servers and clients.

SSL uses X.509 [4] certificates. Server certificates provide a way for clients to authenticate the identity of a server. The client uses the server's public key to negotiate a secure TCP connection with the server. Optionally, the server can authenticate clients by verifying the contents of the clients' certificates.

Even though SSL provides secure communications between servers and clients, it cannot protect against end-system threats [14]. For instance, if a user receives sensitive information from the server over a secure channel, it does not mean that the information is saved securely in the user's machine. In other words, once the user receives the information from the server over the secure channel, he is able to change the information or give it to other people, because SSL does not support security services in the user's end system. However, as we will see later in this paper, SSL can be used as part of our solution to protect information in our implementation.

3. SYSTEM ARCHITECTURE OF SECURE WORKFLOW MANAGEMENT SYSTEM

In this section, we describe the system architecture of our secure workflow management system for dynamic collaboration based on our implementation. There are five major components in the system: design tool, policy server, runtime engine, monitor, and users. Figure 4 shows the components and their relationships in the system. Detailed descriptions about the implementation of this architecture are available in [7, 8].

The *design tool* allows workflow designers to design independent workflows and express their global and local policies and constraints. Global policies and constraints (e.g., User-Role Assignment (URA)) are transferred to the policy server and applied to the whole system. "Global" can be translated from a whole workflow (enterprise) to the whole system, which supports multiple workflows (enterprises). Local policies and constraints (e.g., Permission-Role Assignment (PRA)) are transferred and applied to only relevant tasks in the runtime engine autonomously. Technically, it is always possible to enforce URA locally or

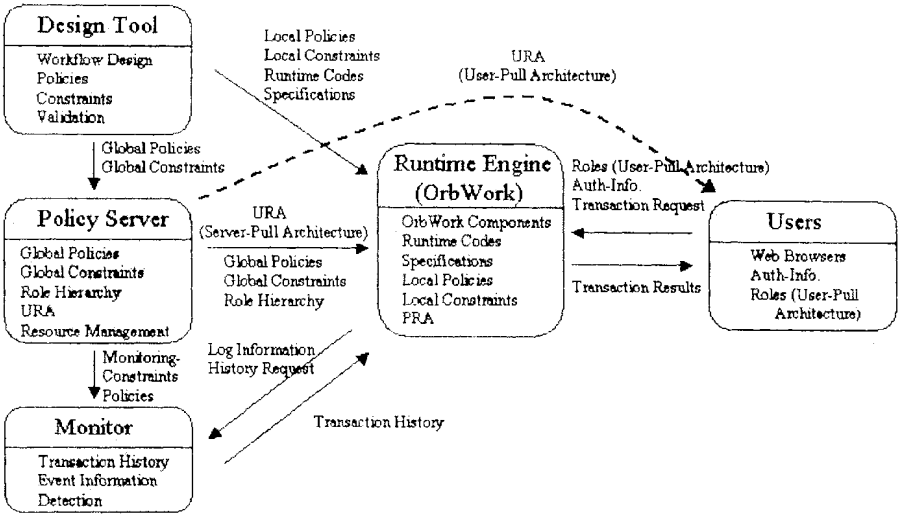


Figure 4 System Architecture of Secure Workflow Management System

PRA globally. Furthermore, different workflows (enterprises) may have different URAs and PRAs. However, we believe that our policy enforcement (global URA and local PRA) is efficient for maintaining organizational consistency and providing autonomy of tasks in the runtime engine. The design tool also validates if a workflow design is consistent and sound. After the designer finishes workflow design, the design tool generates runtime codes and specifications for the workflow that will be used in the runtime engine.

The *policy server* provides global policies and constraints to the other components in the system. For instance, it enforces URA or resource management for the whole system. A typical component in the policy server is a role server. The role server provides role hierarchy and URA information to support RBAC in the system. In the server-pull RBAC architecture (see Section 2.2), individual task servers connect to the role server and pull the user’s roles on demand. In the user-pull RBAC architecture (see Section 2.2), the user connects to this role server and pulls his roles after proper authentication procedures (denoted by a dotted line in Figure 4). Later, he uses those roles in the task servers in the runtime engine to execute human tasks. Technically, a single user-credential can be issued by the policy server and used for both authentication and authorization in the runtime engine. For instance, the policy server can issue an X.509 certificate for the user including the user’s roles and public-key information. Once the user pulls this certificate, he can use it to prove his identity and roles in

the task servers. However, we do not claim that this kind of bundled certificate is always good. Especially, if the lifetimes of a user's role and public-key information are different, or if different authorities must issue the role and identity information, bundled certificate may not be a good solution. Instead, we can use two different certificates to satisfy the above requirements. In this case, we must support the binding of attributes (e.g., roles) and identification for each user [15]. For instance, if Alice presents Bob's roles with her authentication information to the Web server, she must be rejected. It is important to note that the policy server does not have local policies and constraints, which are defined by the design tool and enforced by the individual tasks in the runtime engine.

The *runtime engine* consists of OrbWork (described in Section 2.1) components (task servers, worklist servers, and data servers), PRA, and the information generated by the design tool, such as runtime codes, specifications, local policies, and local constraints. It conducts the workflow tasks, including human tasks and non-human tasks, using the OrbWork components in conjunction with the runtime codes and specifications generated by the design tool. During installation and execution, the runtime engine refers to the PRA, local policies and constraints that it has, and the URA, global policies and constraints that the policy server provides. The runtime engine also refers to the monitor to get the transaction history and make a correct decision. It is important to note that the runtime engine does not have global policies and constraints, which are defined by the design tool and enforced by the policy server.

The *monitor* consists of a monitor server and client. The monitor server receives event information from the runtime engine and records it in a file. The monitor server has application-layer monitoring functions that provide event information, based on its clients' interests. Furthermore, the monitor server provides the transaction history to the runtime engine (if it is necessary) so that the runtime engine can make a correct decision that complies with the policies and constraints based on the user's previous transaction history [2]. Inter-organizational workflows may consist of several autonomous workflows. Hence, there may be multiple monitor servers. In our implementation, there is a monitor server per runtime engine. Each monitor server refers to the policy server for its monitoring policy and constraints, and has its own database so that it can record events from OrbWork and answer any query from OrbWork or monitor clients. Monitor clients can register their topics of interests to monitor servers. For example, one monitor client may be interested in all events in a specific workflow while another monitor client may be interested in only events that have to do with a specific task. The monitor server records clients' interests and dispatches only those events that each client is interested in.

In our system, *users* communicate with the runtime engine using Web browsers via HTTP or HTTPS. Users are assigned to their roles in the policy server (particularly, role server) under the enterprise policy. When a user connects to

the runtime engine using a Web browser, the runtime engine authenticates the user by means of existing authentication mechanisms such as passwords, Kerberos [19], X.509, and so on. In the user-pull RBAC architecture, the user's role information is transferred to the runtime engine from the user's machine (assuming that the user pulled his roles from the role server before). In the server-pull RBAC architecture, the runtime engine pulls the user's role information from the role server after it authenticates the user.

4. SECURITY SERVICES FOR DYNAMIC COLLABORATION

In Section 3, we describe the architecture of our secure workflow management system. Each component in the system may be involved in multiple workflows, which may span multiple organizations. This implies that the complexity of security services for inter-organizational enterprises becomes higher than fully in-house projects. In other words, if more organizations are participating in the enterprise, then more efficient and strong security services are required (sometimes even new security services are required). In this section, we focus on the security services for a secure workflow management system for dynamic collaboration and describe how we have provided those services to our secure workflow systems.

4.1. SECURE COMMUNICATION

Basically, there are two different kinds of communications that we need to protect in our system. Firstly, we need to protect the communications between users and the OrbWork components. Secondly, we need to protect the communications among the OrbWork components. There may be many possible technologies and implementations to satisfy those requirements. Since one of our strategies in this work is the maximum use of available COTS security solutions with the minimum modification of the system components, we have decided to use a standard technology, SSL (described in Section 2.3), for our purposes.

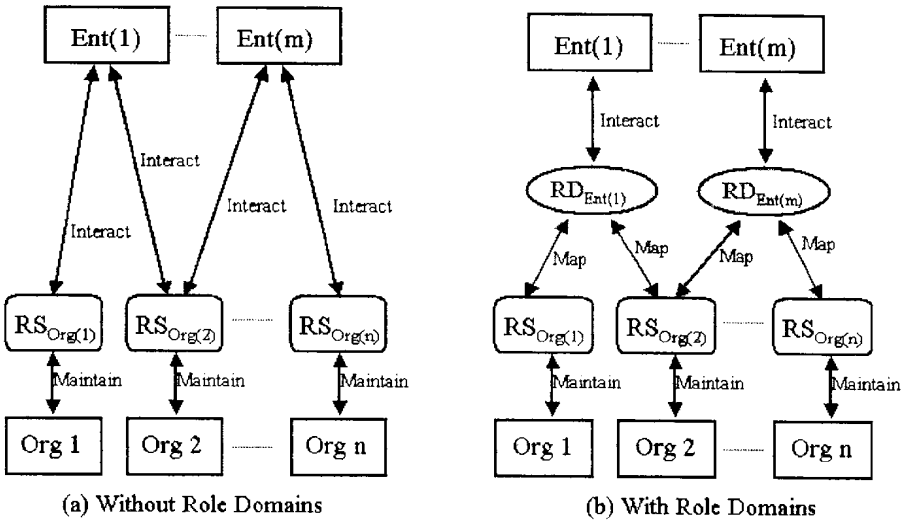
The runtime engine (OrbWork) supports both HTTP and IIOP. The former supports the communications between the OrbWork components and users via their Web browsers. The latter supports the communications among the OrbWork components in CORBA, where all objects access other objects or services via Object Request Brokers (ORBs). By integrating an SSL (we used Phaos' SSLava [20] in our implementation) package with OrbWork, we provide HTTPS for the secure communications between users and the OrbWork components, and SSL-IIOP for the secure communications among the OrbWork components in CORBA. We do not describe other alternative security technologies, such as IPSEC [6], SECIOP (Secure Inter-ORB Protocol) using SPKM (Simple Public-

Key Mechanism [1]), Kerberos [19], or SESAME (Secure European System for Applications in a Multivendor Environment [12]), or DCE-CIOP (Distributed Computing Environment - Common Inter-ORB Protocol) using DCE [17], in detail in this paper, since we believe HTTPS and SSL-IIOP are simple and adequate solutions for our purposes.

4.2. SEPARATING SECURITY INFRASTRUCTURES FOR ORGANIZATIONS AND ENTERPRISES

When several organizations are involved in a large inter-organizational enterprise, especially, when dynamic collaboration is required, there are several security issues that would not be considered in a static in-house project. First, each organization has its own security infrastructure (e.g., organizational role hierarchy), which is different from others including that of the inter-organizational enterprise. If there is a direct assignment between an organizational role and the permission for the inter-organizational enterprise, changes in an organization role hierarchy requires unexpected changes in PRA (Permission-Role Assignment) for the enterprise. Second, the participating organizations may change during the life cycle of the enterprise. For example, a new organization may replace an old organization or there may be a merger or separation among organizations. In this case, how can we assign or revoke users to or from their job responsibilities (e.g., roles) for the enterprise efficiently? To resolve the above problems, we could change the organization security infrastructure (e.g., role hierarchy in the above example) to fit the enterprise's security infrastructure whenever it is necessary. However, it is not sound for dynamic collaboration to restructure each organization's security infrastructure for a particular inter-organizational enterprise. Usually, the lifetime of an enterprise is shorter than those of participating organizations. Furthermore, each organization may support several different enterprises with others. Therefore, we should insulate the security infrastructures for participating organizations and their inter-organizational enterprises.

To achieve this goal, we introduce a concept of role domain, which is a role structure interface for an inter-organizational enterprise. Figure 5 shows two different cases for managing security structures (role structures in this example) for organizations and their inter-organizational enterprises. The relationship between a role domain and the role structures of organizations is similar to an interface in client-server interactions in a distributed environment. It is each organization's responsibility to map its own role structure to the enterprise's role domain. In this case, the role structures of participating organizations can be managed independently and autonomously from those of the enterprises as depicted in Figure 5(b). One organization may map its own role structure to multiple role domains in different ways if it is involved in different



Org (n): Organization n
 RS_{Org(n)}: Role Structure for Organization n
 Ent(m): Enterprise m
 RD_{Ent(m)}: Role Domain for Enterprise m

Figure 5 Managing Security Structures for Organizations and Enterprises

inter-organizational enterprises. The tasks in the individual enterprises require specific roles in its enterprise’s role domain for their access control decisions instead of the users’ roles in their organizations. Detailed descriptions for the access control mechanisms, including fine-grained and context-based access control with dynamic constraints, within the tasks are available in [9].

4.3. PROVIDING DIFFERENT PRIVILEGES TO DIFFERENT USERS

Usually, a large collaborative enterprise spans several organizations, which support a variety of tasks executed by many different users, and consist of many different components, which may change dynamically. It is obvious that individual users - who may belong to different organizations - should have different privileges (roles in our case) for more secure and efficient enterprise management. Therefore, we need to provide different privileges to different users based on the users’ needs-to-do in the enterprise. For example, users working on task T1 need to access the components (tasks) related to T1, but

may not need to access (even know the existence of) the components that are irrelevant to T1 or under the control of other enterprises.

Technically, we could control each user's privileges by the conventional identity-based access control mechanism. This could work for a small project, where a small number of users are involved. However, for a large enterprise, where many users from different organizations join and leave dynamically, the identity-based access control mechanism is inefficient and becomes too complicated to manage. Fortunately, individual users have common job functionalities (abstracted as roles in this paper) for the enterprise. Therefore, we devise a strong and efficient mechanism to provide different privileges to different users by integrating the RBAC model (described in Section 2.2) with our role domain concept (described in Section 4.2), and enforce this mechanism in our system.

In our design tool (see Section 3), we provide a way to specify a required role set (including role domain and roles) for each task in the following format, where RD_m is a specific role domain and R_{mn} is a specific role in the role domain RD_m .

$$[\{ RD_1 : (R_{11} \vee R_{12} \vee \dots \vee R_{1n}) \} \vee \{ RD_2 : (R_{21} \wedge R_{22} \wedge \dots \wedge R_{2n}) \} \vee \dots \vee \{ RD_m : (R_{m1} \vee R_{m2} \vee \dots \vee R_{mn}) \}]$$

Workflow designers specify the required role set for each task in the workflow design tool. This will be enforced by each task during the runtime. For example, if task T1 has a required role set as follows.

$$[\{ \text{SchoolProject} : (\text{teacher} \vee \text{instructor}) \} \vee \{ \text{CompanyProject} : (\text{manager} \wedge \text{staff}) \}]$$

A user who has the teacher, instructor, or senior roles (to teacher or instructor) in the SchoolProject role domain, or the manager and staff, or senior roles (to manager and staff) in the CompanyProject role domain is allowed to execute the task T1 and access the components or other tasks related to T1. Basically, the access control and the level of the services are based on the user's assigned roles in the enterprise's role domain. The required role set does not consider the user's organization or identity.

4.4. DESIGN VALIDATION

Since several portions of a workflow design may be assembled to accomplish an enterprise level mission, it is important to validate that the overall design is consistent and sound. We provide translators for converting an inter-organizational workflow design into inputs to an existing Petri-net based analysis tool, Woflan [22], and a model checking tool, Spin [3], so that the consistency of the inter-organizational workflow design can be validated. Detailed mecha-

nisms of design validation and related examples will be described in our future publications.

5. IMPLEMENTATION SUMMARY AND CONCLUSION

We have developed a GUI (Graphical User Interface)-based workflow design tool (described in Section 3) in JAVA. The design tool is integrated with Woflan and Spin for design validation. Currently, we are using modified OrbWork (see Section 2.1) as our runtime engine, which uses IONA's JAVA implementation of Orbix ORBs (Object Request Brokers [11]) version 3 to support CORBA in the system. To accommodate secure collaboration, OrbWork has to be extended in two major areas. The first area is to support the extended workflow interoperability model (we call it cooperative processes model) that we introduced in [8]. The second area is the incorporation of SSL that supports secure communications between clients and servers. We have integrated Phaos' SSLava [20] version 1.11 with OrbWork to provide secure communications. To support monitor functions (described in Section 3) in the system, we use MS Access via JDBC data access API [5] to store and provide transaction histories.

In this paper, we have addressed the security services for a secure workflow system to support dynamic collaboration; providing secure communications between users and system components, separating security infrastructures for organizations and their enterprises, providing different privileges to different users, and validating workflow designs. We have convinced why we need these services and described how we implemented them in our secure workflow management system. Although we have applied our ideas to particular technologies, such as workflows and RBAC, in this paper, we believe it is always possible to apply our approaches to other security systems, which support many users from different organizations.

References

- [1] C. Adams. *The Simple Public-Key GSS-API Mechanism (SPKM)*. RFC 2025, October 1996.
- [2] G. Edjlali, A. Acharya, and V. Chaudhary. *History-based Access Control for Mobile Code*. In Proceedings of the 5th ACM Conference on Computer and Communications Security, San Francisco, CA, November 1998.
- [3] G.J. Holzmann. *The Model Checker Spin*. IEEE Transactions on Software Engineering, Vol. 23, No.5, pp. 279-295, May 1997. See also <http://cm.bell-labs.com/cm/cs/what/spin/>.
- [4] ITU-T Recommendation X.509. *Information Technology - Open Systems Interconnection - The Directory: Authentication Framework*. International

Telecommunication Union Standardization Sector, 1997.

- [5] *JDBC Data Access API*. <http://java.sun.com/products/jdbc/>, Sun Microsystems, 2000.
- [6] S. Kent and R. Atkinson. *Security Architecture for the Internet Protocol*. RFC 2401, November 1998.
- [7] M. H. Kang, B. J. Eppinger, and J. N. Froscher. *Tools to Support Secure Enterprise Computing*. In Proceedings of 15th Annual Computer Security Applications Conference, Phoenix, Arizona, December 1999.
- [8] M. H. Kang, J. N. Froscher, A. P. Sheth, K. J. Kochut, and J. A. Miller. *A Multilevel Secure Workflow Management System*. In Proceedings of the 11th Conference on Advanced Information Systems Engineering, Heidelberg, Germany, June 1999.
- [9] M. H. Kang, J. S. Park, and J. N. Froscher. *Access Control Mechanisms for Inter-organizational Workflow*. In Proceedings of 6th ACM Symposium on Access Control Model and Technologies (SACMAT), Chantilly, Virginia, May 2001.
- [10] K. Kochut, A. Sheth, and J. Miller. *ORBWork: A CORBA-Based Fully Distributed, Scalable and Dynamic Workflow Enactment Service for META-TEOR*. UGA-CS-TR-98-006, Technical Report, Department of Computer Science, University of Georgia, 1998.
- [11] *Orbix 3*. http://www.iona.com/products/orbix3_home.htm, IONA, 2000.
- [12] T. Parker and D. Pinkas. *SESAME V4 - Overview*. SESAME Technology, December 1995.
- [13] J. S. Park and R. S. Sandhu. *Smart Certificates: Extending X.509 for Secure Attribute Service on the Web*. In Proceedings of 22nd National Information Systems Security Conference (NISSC), Crystal City, Virginia, October 1999.
- [14] J. S. Park and R. S. Sandhu. *Secure Cookies on the Web*. IEEE Internet Computing, Vol. 4, No. 4, pp. 36-44, July-August 2000.
- [15] J. S. Park and R. S. Sandhu. *Binding Identities and Attributes Using Digitally Signed Certificates*. In Proceedings of 16th Annual Computer Security Applications Conference (ACSAC), New Orleans, Louisiana, December 2000.
- [16] J. S. Park, R. S. Sandhu, and G. J. Ahn. *Role-based Access Control on the Web*. ACM Transactions on Information and System Security (TISSEC), Vol. 4, No. 1, February 2001.
- [17] W. Rosenberry, D. Kenney, and G. Fisher. *Understanding DCE*. O'Reilly & Associates, 1992.

- [18] R. S. Sandhu, E. J. Coyne, H. Feinstein, and C. Youman. *Role-based Access Control Models*. IEEE Computer, Vol. 29, No. 2, pp. 38-47, February 1996.
- [19] J.F. Steiner, C. Neuman, and J.I. Schiller. *Kerberos: An Authentication Service for Open Network Systems*. In Proceedings of Winter USENIX Conference, 1988.
- [20] *Phaos SSLava*. http://www.phaos.com/e_security/prod-ssl.html, Phaos Technology, 2000.
- [21] D. Wagner and B. Schneir. *Analysis of the SSL3.0 Protocol*. In Proceedings of the 2nd UNIX Workshop on Electronic Commerce, November 1996.
- [22] H.M.W. Verbeek, T. Basten, and W.M.P. van der Aalst. *Diagnosing Workflow Processes Using Woflan*. Computing Science Report 99/02, Eindhoven University of Technology, Eindhoven, 1999.

This Page Intentionally Left Blank

ON SECURELY SCHEDULING A MEETING

Thomas Herlea, Joris Claessens, Bart Preneel
COmputer Security and Industrial Cryptography (COSIC)
Dept. of Electrical Engineering – ESAT
Katholieke Universiteit Leuven
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
firstname.lastname@esat.kuleuven.ac.be
<http://www.esat.kuleuven.ac.be/cosic/>

Gregory Neven, Frank Piessens, Bart De Decker
Dept. of Computer Science
Katholieke Universiteit Leuven
Celestijnenlaan 200A, B-3001 Leuven-Heverlee, Belgium
firstname.lastname@cs.kuleuven.ac.be
<http://www.cs.kuleuven.ac.be/cwis/research/distrinet/>

Abstract When people want to schedule a meeting, their agendas must be compared to find a time suitable for all participants. At the same time, people want to keep their agendas private. This paper presents several approaches which intend to solve this contradiction. A custom-made protocol for secure meeting scheduling and a protocol based on secure distributed computing are discussed. The security properties and complexity of these protocols are compared. A trade-off between trust and bandwidth requirements is shown to be possible by implementing the protocols using mobile agents.

Keywords: mobile agents, secure distributed computation, meeting scheduling

1. INTRODUCTION

When negotiating meetings, the participants look up, communicate and process information about each other's agendas trying to find a moment when they are all free to attend the meeting. Due to the private nature of a person's schedule, as little as possible should be revealed to any other party during that

negotiation. Ideally, only the result of the negotiation should be known to the participants (and to the participants only), and any other information about the users' agendas should remain secret.

An easy solution for scheduling a meeting is to broadcast the schedules to all participants, but this totally neglects the privacy of the participants' agendas. Another solution is to send all schedules to a trusted third party, but finding one such single third party trusted by every participant, will be very difficult in practice.

Some existing meeting scheduling applications, like for example "Yahoo! Calendar", define access levels for viewing and modifying agenda entries, and define user groups to which these access levels are assigned. This is only necessary because the comparison between schedules must be done by the users themselves. Our approaches eliminate the need for managing access control, as they are not based on users directly accessing each other's agenda.

This paper presents more secure solutions. Their goal is for participants to be able to negotiate a meeting whereby parties have no direct access to each other's agenda, whereby parties do not rely on another party for telling the final result, and whereby no information about the agendas is revealed, but the final result, i.e., the particular time the meeting can be scheduled, or the fact that the meeting cannot be scheduled.

This paper builds on the work done in [6] and [3] and shows the trade-offs that can be made in security, level of trust, and efficiency, when choosing a particular negotiation protocol and a specific implementation approach.

The paper is organized as follows. Section 2 presents a custom-made negotiation protocol. Section 3 presents an alternative approach based on secure distributed computing. Both approaches are analyzed from a security and complexity point of view. Section 4 discusses the use of mobile agents for secure meeting scheduling, and presents the "agenTa" prototype implementation. We conclude in Sect. 5.

2. USING A CUSTOM-MADE NEGOTIATION PROTOCOL

2.1. DATA REPRESENTATION

There exists a representation which reduces the problem of deciding if the meeting can be scheduled at a certain moment to a logical AND operation.

As shown in Fig. 1, an agenda will be represented as a bit string in the following way: for each time slot in the schedule, there is one bit indicating whether the negotiator can (1) or cannot (0) attend a meeting of the specified length which would start at that time. The finer the granularity and the longer the negotiation window, the more bits there will be in the representation.

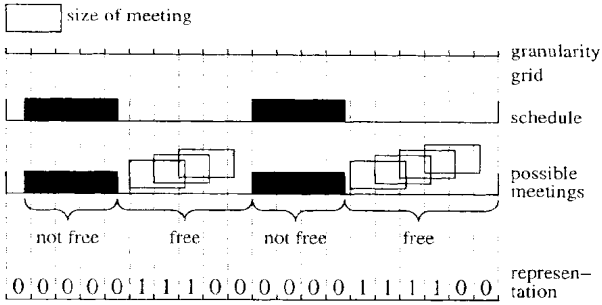


Figure 1 Conversion from agenda to representation

2.2. SCHEDULING MODEL

In our model, a meeting scheduling starts with an invitation phase. The initiator broadcasts to the invitees a set of negotiation parameters such as meeting length, negotiation window (limited time span in which to attempt the meeting scheduling) and a complete list of invitees. Each invitee broadcasts to all others a reply indicating whether it will accept or decline the negotiation invitation. Because broadcasts are used, no invitee can be misled as to the set of negotiators it will encounter in the second phase.

In the second phase, called negotiation, the negotiators try the time slots one by one and attempt to schedule the meeting. For each time slot the negotiation takes place according to the protocol outlined below. If the meeting was successfully scheduled the negotiators move on to the third phase, otherwise the next time slot is tried. After independently arriving to a result concerning a certain time slot, each participant broadcasts the result to the others and checks whether all results coincide. This allows for detection of partial failures and attacks which try to mislead a subset of the negotiators.

In the third phase either the common result is presented to the users, or the users are informed that no meeting can take place. If there is a common result, users might confirm their commitment to the scheduled time on a separate channel (e-mail, telephone), independently of the scheduling process.

2.3. SCHEDULING A MEETING

For the purpose of this subsection we will refer to the representation of an agenda according to the description in the previous subsection as “schedule.”

Instead of comparing schedules, the negotiation should be based on comparing protected forms of the schedules. The schedules are protected in a way which still allows scheduling to be performed by broadcasting the protected

forms to all negotiators and letting them process the data without fear of the unprotected form to be revealed.

The binary XOR operation between the schedule and a mask is a transformation which still allows scheduling to be performed in the sense that the (in)equality of two or more bits is preserved when they are all XORed with the same mask.

If all negotiators know the mask, they are able to retrieve the original schedules easily, by unmasking the broadcasted data. The solution is to let the mask be a shared secret, that is, all negotiators will contribute when building it, but it will not be revealed to any of them.

The negotiation protocol then goes as follows:

- 1 In step one of the negotiation protocol, each negotiator chooses a random mask, and XORs it with its schedule. This random mask is actually a partial mask. The shared secret will be the XOR of all partial masks, and is called global mask. Even if only one negotiator keeps its partial mask secret, the others cannot find the global mask solely using their partial masks.
- 2 In step two of the protocol, all schedules visit all negotiators exactly one time. At each visit they are masked with the partial mask of that particular negotiator. In the end, all original schedules are thus masked with the global mask, without the need for the negotiators to disclose their partial mask. Since the schedule is first masked with its owner's partial mask it remains secret during its visits.

A negotiator must be unable to identify a protected schedule as representing its own schedule: otherwise performing XOR between the original and the protected schedule reveals the global mask, allowing the negotiator to retrieve all original schedules. Therefore during the trip to all negotiators, the schedule must be forwarded randomly between the negotiators in order to make it impossible to trace. The schedule must have attached a list of negotiators it hasn't visited yet, decremented at each forwarding, in order to prevent multiple maskings with the same partial mask.

Note that for countering attempts to trace a schedule by attackers who have a global view on the network, all communications should be encrypted.

- 3 In step three, all protected schedules are broadcasted. Each negotiator looks independently for a time slot when all protected schedules have the same value. That implies that the original schedules are identical, too, for that time slot but does not provide any clue whether the negotiators are free or busy for that time slot. The clue is provided by each negotiator's

schedule for that time slot. If the negotiator is free then, it means all negotiators are free then and the meeting can be scheduled. For time slots when some are busy and some are free, it is not possible to figure out who are the busy ones and who are the free ones.

Note that our scheduling protocol does not specify any form of negotiator authentication. This is however needed for linking the protocol messages to their originators. Depending on the meeting application, the desired form of authentication can be added to the protocol.

Figure 2 shows the negotiation protocol as performed by three parties. For easy understanding of the protocol the schedules in the simulation are following the same route and the maskings appear to be performed simultaneously by the three negotiators. In reality the process is asynchronous (some negotiators may be idle while others are masking) and routing is random (in the end some negotiators may have nothing to broadcast while others may broadcast several protected schedules). Another difference is that in reality only one bit is processed at a time (otherwise an attack is possible, see following section). If the meeting can be scheduled in the corresponding time slot the protocol stops, otherwise the next time slot is processed.

2.4. SECURITY ANALYSIS

Our custom-made protocol does not require one single entity to be trusted. It however does not completely protect the privacy of the participants' agenda, as attacks by both passive and active adversaries are possible.

Bad slots. There may be time slots for which all users are busy and therefore all protected slots will be equal. By checking against the original schedule each negotiator will avoid scheduling a meeting in that slot but it will also know everybody else's schedule for that slot (i.e., everybody is busy). Because they constitute an infringement on all users' privacy we call these slots *bad slots*.

Entropy attack. The reason for performing the negotiation one slot at a time is to prevent the following attack. If the negotiation is done on sequences of slots, when all the broadcasted masked schedules are received, it still is possible for a party to recognize its original schedule. It can be done by testing all the masks which transform the original schedule into one of the protected forms. The correct global mask can be recognized by the fact that by unmasking the other protected schedules with it, bit strings are obtained which have the entropy expected from a schedule.

Negotiating one bit at a time, with fresh partial masks for each bit and stopping when a meeting is scheduled counters this attack because each mask bit and schedule bit have maximal entropy.

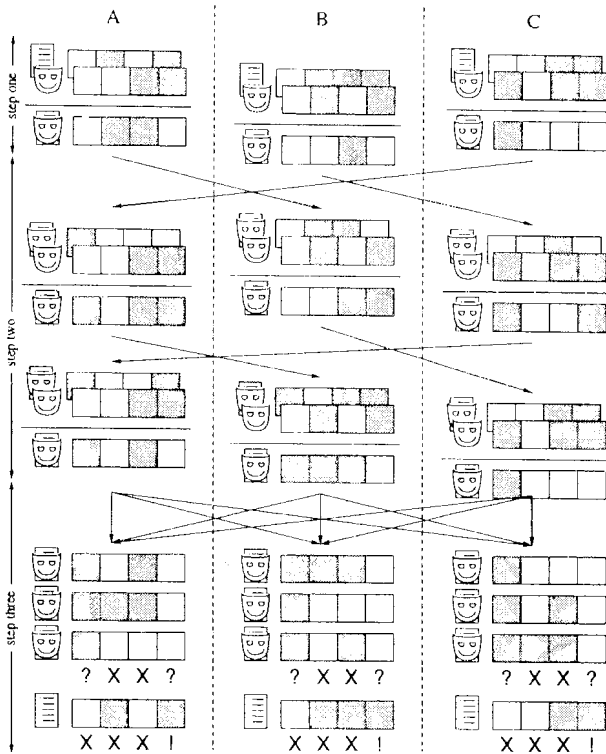


Figure 2 Simulation for three negotiators

Number of parties. When only two parties are negotiating, each can deduce the schedule of the other based on their own schedule and the comparison between the protected forms of the schedules. Besides that, the global mask is straightforward to find because the original schedule can be linked to its protected form. Also when only three or four parties are negotiating it is sometimes possible to find out the global mask by tracing back schedules. For five or more participants the ability to trace a schedule along its route decreases as the number of participants increases.

Dummy negotiators could be introduced to artificially increase the number of parties, and thus to alleviate this problem. In a broadcasting communication environment encrypted dummy messages could also be sent to make the real schedules untraceable.

Rogue negotiators. Active adversaries could attack the protocol in various ways.

A simple denial of service attack can be mounted by negotiating based on a fully busy schedule instead of declining the invitation. Since the protocol relies on the negotiators consistently using their partial mask, the protocol has unpredictable outcomes if a negotiator randomly changes its partial mask during the negotiation of a time slot.

Goal-oriented misbehavior is also possible. A negotiator can wait to be the last to broadcast the protected schedule(s) it has. This way it is able to detect first when a meeting could take place. In that case it can broadcast a false protected form, preventing the meeting from being scheduled. It knows everybody else's schedule for that time slot, while the others do not.

2.5. COMPLEXITY

For analyzing the complexity of the scheduling we count the messages that are sent between the negotiators. In a distributed environment it is expected that sending messages will be much more resource consuming than masking or a comparison between bits. Since much of the processing is done in parallel, bandwidth is more important. Remember that the negotiation protocol is performed bit by bit.

Note that for n negotiators a broadcast is of complexity $n - 1$. When an all-to-all broadcast is needed it has complexity $n(n - 1)$.

The scheduling starts with a simple broadcast of the invitation. $C_1 = n - 1$. All negotiators (except for the initiator) must announce their position towards the invitation. These broadcasts adds complexity $C_2 = (n - 1)(n - 1)$. For getting masked, one bit must visit all negotiators and then be broadcasted: $2(n - 1)$. This happens to each negotiator's bit in a round: $C_3 = 2n(n - 1)$. If the number of bits in a schedule is l , after at most l rounds the protocol will end. In the check phase of the scheduling, all negotiators broadcast their result or the fact that no meeting could be scheduled to all others: $C_4 = n(n - 1)$. Note that only positive results (i.e., a meeting is possible) are broadcasted. If the result is negative, the agents automatically go to the next bit. If the result is still negative after the last bit, it was not possible to schedule a meeting.

Therefore at most $C = C_1 + C_2 + lC_3 + C_4 = (1 + n - 1 + 2nl + n)(n - 1) = (2 + 2l)n(n - 1)$ messages are sent. For example, for a scheduling window of 3 eight-hour working days, granularity 1 hour ($l = 24$) and 5 participants ($n = 5$) this amounts to at most 1000 messages; for 10 participants in the same conditions, there will be up to 4500 messages sent.

3. USING SECURE DISTRIBUTED COMPUTING

3.1. THE PROBLEM OF SECURE DISTRIBUTED COMPUTING

Usually, the problem of Secure Distributed Computing (SDC) is stated as follows. Let f be a publicly known function taking n inputs, and suppose there are n different parties, each holding their own private input x_i ($i = 1 \dots n$). The n parties want to compute the value $f(x_1, \dots, x_n)$ without leaking any information about their private inputs to the other parties (except of course the information about x_i that is implicitly present in the function result). In descriptions of solutions to the Secure Distributed Computing problem, the function f is usually encoded as a boolean circuit, and therefore Secure Distributed Computing is also often referred to as *secure circuit evaluation*.

Over the past two decades, a fairly large variety of solutions (other than the trivial one using a trusted third party) to the problem has been proposed. An overview is given by Franklin [4] and more recently by Cramer [2].

3.2 HOW TO PERFORM GENERAL SECURE DISTRIBUTED COMPUTING

The core problem of SDC is that we want to perform computations on hidden data (using encryption, secret sharing or other techniques) without revealing the data. One class of techniques to compute with encrypted data is based on *homomorphic probabilistic encryption*. An encryption technique is *probabilistic* if the same cleartext can encrypt to many different ciphertexts under the same encryption key. To work with encrypted bits, probabilistic encryption is essential, otherwise only two ciphertexts (the encryption of a zero and the encryption of a one) would be possible, and cryptanalysis would be fairly simple. An encryption technique is *homomorphic* if it satisfies at least one equation of the form $E(x \text{ op } y) = E(x) \text{ op}' E(y)$ for some operations **op** and **op'**. A homomorphic encryption scheme allows operations to be performed on encrypted data, and hence is suitable for secure circuit evaluation.

In [5], Franklin and Haber present a protocol that evaluates a boolean circuit on data encrypted with such a homomorphic probabilistic encryption scheme. In order to support any number of participants, they use a *group oriented* encryption scheme, i.e., an encryption scheme that allows anyone to encrypt, but that needs the cooperation of all participants to decrypt. In the group oriented encryption scheme used by Franklin and Haber, a bit b is encrypted for a group of participants $S \subseteq \{1 \dots n\}$ as

$$E_S(b) = \left[g^r \pmod N, (-1)^b \left(\prod_{j \in S} g^{K_j} \right)^r \pmod N \right]$$

where $N = pq$, p and q are two primes such that $p \equiv q \pmod{4}$, and $r \in_R Z_N$. The public key is given by $[N, g, g^{K_1} \pmod{N}, \dots, g^{K_n} \pmod{N}]$, while K_i is the private key of the i th participant. This scheme has some additional properties that are used in the protocol:

- *XOR-Homomorphic.* Anyone can compute a joint encryption of the XOR of two jointly encrypted bits. Indeed, if $E_S(b) = [\alpha, \beta]$ and $E_S(b') = [\alpha', \beta']$, then $E_S(b \oplus b') = [\alpha\alpha' \pmod{N}, \beta\beta' \pmod{N}]$.
- *Blindable.* Given an encrypted bit, anyone can create a random ciphertext that decrypts to the same bit. Indeed, if $E_S(b) = [\alpha, \beta]$ and $r \in_R Z_N$, then $[\alpha g^r \pmod{N}, \beta (\prod_{j \in S} g^{K_j})^r \pmod{N}]$ is a joint encryption of the same bit.
- *Witnessable.* Any participant can withdraw from a joint encryption by providing the other participants with a single value. Indeed, if $E_S(b) = [\alpha, \beta]$, it is easy to compute $D_i(E_S(b))$ from $W_i([\alpha, \beta]) = \alpha^{-K_i} \pmod{N}$

First of all, the participants must agree on a value for N and g , choose a secret key K_i and broadcast $g^{K_i} \pmod{N}$ to form the public key. To start the actual protocol, each participant broadcasts a joint encryption of his own input bits. To evaluate an XOR-gate, everyone simply applies the XOR-homomorphism. The encrypted output of a NOT-gate can be found by applying the XOR-homomorphism with a default encryption of a one, e.g. $[1, -1 \pmod{N}]$.

The encryption scheme is not AND-homomorphic, so the evaluation of an AND-gate will be more troublesome. Suppose the encrypted input bits for the AND-gate are $\hat{u} = E(u)$ and $\hat{v} = E(v)$. To compute a joint encryption $\hat{w} = E(w) = E(u \wedge v)$, they proceed as follows:

- 1 Each participant i chooses random bits b_i and c_i and broadcasts $\hat{b}_i = E(b_i)$ and $\hat{c}_i = E(c_i)$.
- 2 Each participant repeatedly applies the XOR-homomorphism to calculate $\hat{u}' = E(u') = E(u \oplus b_1 \oplus \dots \oplus b_n)$ and $\hat{v}' = E(v') = E(v \oplus c_1 \oplus \dots \oplus c_n)$. Each participant broadcasts decryption witnesses $W_i(\hat{u}')$ and $W_i(\hat{v}')$.
- 3 Everyone can now decrypt \hat{u}' and \hat{v}' . By repeatedly applying the fact that $(a \oplus b) \wedge c = (a \wedge c) \oplus (b \wedge c)$, one can prove that $w' = u' \wedge v' = (u \wedge v) \oplus w_1 \oplus \dots \oplus w_n$ where $w_i = (u \wedge c_i) \oplus (b_i \wedge c_1) \oplus \dots \oplus (b_i \wedge c_n) \oplus (b_i \wedge v)$. Each participant is able to compute a joint encryption of w_i : he knows b_i and c_i (he chose them himself) and he received encryptions \hat{c}_j from the other participants, so he can compute $E(b_i \wedge c_j)$ as follows: if $b_i = 0$, then $b_i \wedge c_j = 0$, so any default encryption for a zero will do, e.g. $[1, 1]$.

Otherwise, if $b_i = 1$, then $b_i \wedge c_j = c_j$, so \hat{c}_j is a valid substitution for $E(b_i \wedge c_j)$.

$E(u \wedge c_i)$ and $E(v \wedge b_i)$ can be computed in an analogous way. He uses the XOR-homomorphism to combine all these terms, blinds the result and broadcasts this as \hat{w}_i .

- 4 Each participant combines \hat{w}' and \hat{w}_j ($j = 1 \dots n$), again using the XOR-homomorphism, to form $\hat{w} = E(w)$.

When all gates in the circuit have been evaluated, every participant has a joint encryption of the output bits. Finally, the participants broadcast decryption witnesses for the output bits to reveal them.

3.3. SECURE MEETING SCHEDULING USING SDC

We already showed how to reduce the problem of scheduling a meeting for n secret agendas to a series of logical AND operations on n secret bits. For every time slot in the schedule, each negotiator has one secret input bit: a one if he is available to start the meeting at that time, a zero if he isn't. Because the Secure Distributed Computing protocol we just discussed can only handle binary gates, we implement the n -ary AND operation as a $\log_2(n)$ -depth tree of binary AND-gates. The output bit of the circuit indicates if this slot is an appropriate starting time for the meeting (1) or not (0).

3.4. SECURITY ANALYSIS

Franklin and Haber show that their protocol is provably secure against passive adversaries (i.e., adversaries who follow the rules of the protocol, but who try to learn as much information from the communication as possible), given that ElGamal encryption with a composite modulus is secure. This means that under the assumption of passive adversaries, complete privacy of all agendas is guaranteed (except of course for the fact that everybody is available at the time the meeting is scheduled). However, the proof Franklin and Haber give uses a more complicated encryption scheme and they mention the one we used here as an alternative. To the best of our knowledge, the security of this encryption scheme is still an open problem.

The protocol is not provably secure against active adversaries (who can deviate from the protocol). For example, a malicious participant can flip the output of an AND gate by XORing his \hat{m}_i with the encryption of a one. For this particular application however, the most obvious attacks don't seem to give rise to substantial information leaks. The SDC protocol presented by Chaum, Damgård and van de Graaf in [1] provides provable security against active adversaries at the cost of higher bandwidth requirements.

3.5. COMPLEXITY

Let's have a closer look at the message complexity of this protocol. The same public and private keys can be used for every evaluation. This means that the initiator's invitation message can contain N and g ($C_1 = n - 1$ messages), while g^{K_i} can be wrapped together with the message that announces each participant's position towards the invitation ($C_2 = (n - 1)(n - 1)$ messages).

The evaluation of a single AND gate consists of four phases, of which the first three need an all-to-all broadcast (consuming $n(n - 1)$ messages each) while the last one doesn't need any communication. Since the AND gates within one level of the tree can be evaluated in parallel, the evaluation of the entire circuit takes $C_3 = \lceil \log_2(n) \rceil \cdot 3n(n - 1)$ messages. The broadcast of the encrypted input bits of the circuit and the broadcast of decryption witnesses for the output bit both take another $C_4 = n(n - 1)$ messages.

If l slot evaluations are needed before a suitable meeting time is found, the total message complexity is given by $C_1 + C_2 + l(C_3 + 2C_4) = n(n - 1)(1 + l(2 + 3 \lceil \log_2(n) \rceil))$. If we consider the same example as we did in the previous section ($l = 24$), this amounts to 5300 messages for 5 participants and 30330 messages for 10 participants.

Before comparing this result to that of the custom-made protocol in the previous section, we should notice that only the number of messages is taken into account, not their size. As $|N|$ should be about 1024 bits to be secure, the messages in the SDC protocol will be larger than the messages in the custom-made protocol. However, since the maximum message length for 10 participants is only 2.5 KB (which easily fits into a single IP packet), we considered the number of transmitted messages more relevant than the number of bits that are strictly needed.

It should also be noted that we do not take into account computation or memory overhead for the protocols. The amount of computation and storage needed for the SDC protocol is considerably higher than for the custom-made protocol.

4. USING MOBILE AGENTS

In this section, it will be shown how mobile agents can be used to reduce the communication overhead of the two solutions for the agenda scheduling problem. The basic idea is to use mobility to bring agents of the participants closer together. Of course, a mobile agent needs to trust his execution platform but we will show that the trust requirements are less strong than for a classical trusted third party (TTP) solution for the meeting scheduling problem.

To compare the trust requirements of the different approaches, we use the following simple trust model. We say a participant *trusts* an execution site if it believes that: (1) the execution site will correctly execute any code sent to it

by the participant; (2) the execution site will correctly (i.e., as expected by the participant) handle any data sent to it by the participant. It also implies that the execution site will maintain the privacy of the data or the code if this is expected by the participant. If p trusts E , we denote this as shown in Fig. 3.



Figure 3 Notation for “ p trusts E ”

To compare bandwidth requirements (for communication overhead), we make the following simple distinction. *High* bandwidth is required to execute one of the discussed protocols. *Low* bandwidth suffices to transmit data or agent code. Also intermittent connections (e.g. for devices that are sometimes disconnected from the network) are considered low bandwidth. We assume low bandwidth communication is available between any two parties. If high bandwidth communication is possible between E_i and E_j , we denote this as shown in Fig. 4.

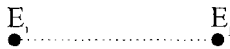


Figure 4 Notation for high bandwidth connection between E_i and E_j

Based on these simple models of communication and trust, we compare three options for implementing secure meeting scheduling.

4.1. A TRUSTED THIRD PARTY

The first, perhaps most straightforward option, is to use a globally trusted third party. Every participant sends its agenda to the TTP who will compute an appropriate meeting time and disseminate the result to the participants. Of course, data must be sent to the TTP, through an authenticated and safe channel. This can be accomplished via conventional cryptographic techniques.

It is clear that this approach has a very low communication overhead: the data is only sent once to the TTP; later, every participant receives the result of the computation. However, every participant should unconditionally trust the TTP. For the case of 4 participants, the situation is as shown in Fig. 5.

It is not clear whether n distrustful participants will easily agree on one single trustworthy third party. This requirement of one single globally trusted execution site is the main disadvantage of this approach.

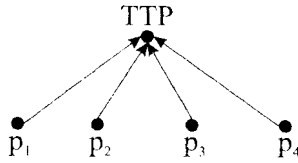


Figure 5 Situation with 4 participants and a TTP.

4.2. CRYPTOGRAPHIC SECURE MEETING SCHEDULING

The second option is the use of cryptographic techniques (as discussed in previous sections) that make the use of a TTP superfluous.

The trust requirements are really minimal: every participant only trusts its own execution site.

Although this option is very attractive, it should be clear from the previous sections that the communication overhead might be too high to be practically useful in a general networked environment. High bandwidth is required between all of the participants. For the case of 4 participants, the situation can be summarized as shown in Fig. 6.

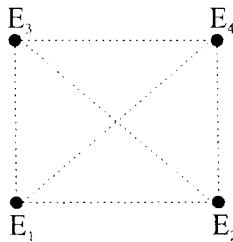


Figure 6 Situation with 4 participants without a TTP.

4.3. USING MOBILE AGENTS

Finally, a third solution tries to combine the two previous options: the communication overhead is remedied by introducing semi-trusted execution sites and mobile agents.

In this approach, every participant p_i sends its representative, agent a_i , to a trusted execution site E_j . The agent contains a copy of the agenda and is capable of running a secure meeting scheduling protocol.

It is allowed that different participants send their agents to different sites. The only restriction being that the sites should be located closely to each other, i.e., should have high bandwidth communication between them.

The amount of long distance communication is moderate: every participant sends its agent to a remote site, and receives the result from its agent. The agents use a cryptographic protocol, which unfortunately involves a high communication overhead. However, since the agents are executing on sites that are near each other, the overhead of the protocol is acceptable. For a situation with 4 participants, we could have the situation as depicted in Fig. 7.

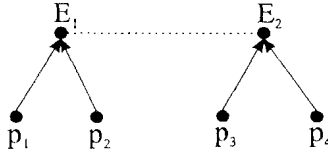


Figure 7 Situation with 4 participants using mobile agents

No high bandwidth communication between the participants is necessary, and there is no longer a need for one single trusted execution site.

4.4. CURRENT IMPLEMENTATION WITH AGLETS

“agenTa” is the name of our prototype implementation of a secure meeting scheduling system. Currently it uses the custom-made protocol described in this paper.

We have used the Aglets SDK 1.1 beta 3 [7], a mobile agents system development kit which was released to the open source community by its creator, IBM. The SDK contains an agent server, the API needed to write agents in Java (called aglets), examples and documentation. The prototype implementation of agenTa has around 3500 lines of Java code.

For the inter-agent communication KQML (Knowledge Query and Manipulation Language) was chosen. KQML was developed at the University of Baltimore Maryland County, and enhanced with security capabilities in [8].

In our implementation, each user’s scheduling application is modular, the user interface, the agenda management and the negotiation being performed by distinct intercommunicating aglets. Only the negotiator aglets of all users take advantage of their mobility to gather on a host where they carry out the negotiation protocol by local communication.

There are no language limitations for implementing the custom-made protocol. Communication relies on transmitting character strings. Therefore, agents implemented with other agent platforms and in other programming languages can take part in the negotiation, provided the platforms can interoperate.

5. CONCLUSION

This paper has shown that there exist several techniques for secure meeting scheduling. Moreover, a trade-off can be made between the level of security that can be obtained, the degree of trust that is required, and the amount of overhead that is caused by the protocol.

When a TTP is used, a meeting can be scheduled very efficiently. The custom-made protocol has more overhead, but does not require trust in a third party. An SDC protocol is more secure than our custom-made protocol, but it is also much less efficient.

Using mobile agents when implementing any protocol can improve the efficiency, while still avoiding the need for one single trusted entity.

Acknowledgements

This work was supported in part by the FWO-Vlaanderen project G.0358.99 and by the Concerted Research Action (GOA) Mefisto-666. Joris Claessens is funded by a research grant of the Flemish Institute for the Promotion of Industrial Scientific and Technological Research (IWT), Gregory Neven is an FWO-Vlaanderen Aspirant, and Frank Piessens is an FWO-Vlaanderen Post-doctoral fellow.

References

- [1] D. Chaum, I. Damgård, J. van de Graaf, "Multiparty computations ensuring privacy of each party's input and correctness of the result." In C. Pomerance, ed., *Advances in Cryptology—CRYPTO '87 Proceedings*, Lecture Notes in Computer Science, LNCS 293, pp. 87–119, Springer-Verlag, New York, 1988.
- [2] R. Cramer, "An introduction to secure computation." In I. Damgård, ed., *Lectures on Data Security*, Lecture Notes in Computer Science, LNCS 1561, pp. 16–62, 1999.
- [3] B. De Decker, F. Piessens, E. Van Hoeymissen, G. Neven, "Semi-trusted Hosts and Mobile Agents: Enabling Secure Distributed Computations." In E. Horlait, ed., *Mobile Agents for Telecommunication Applications*, Lecture Notes in Computer Science, LNCS 1931, pp. 219–232, Springer-Verlag, 2000.
- [4] M. Franklin, "Complexity and security of distributed protocols." Ph.D. thesis, Computer Science Department of Columbia University, New York, 1993.
- [5] M. Franklin and S. Haber, "Joint encryption and message-efficient secure computation ." *Journal of Cryptology*, 9(4), pp. 217–232, Autumn 1996.

- [6] T. Herlea, J. Claessens, D. De Cock, B. Preneel, J. Vandewalle, "Secure Meeting Scheduling with agenTa." *Proceedings of IFIP Communications and Multimedia Security*, 2001.
- [7] Danny B. Lange, Mitsuru Oshima, "Mobile agents with Java: The Aglet API." <http://www.genmagic.com/asa/danny/Wwwj.pdf>.
- [8] Chelliah Thirunavukkarasu, Tim Finin, James Mayfield, "Secret Agents - A Security Architecture for the KQML Agent Communication Language." <http://www.cs.umbc.edu/kqml/papers/secret.ps>.

MODELING AND ANALYZING SEPARATION OF DUTIES IN WORKFLOW ENVIRONMENTS

Konstantin Knorr and Henrik Stormer
Department of Information Technology
University of Zurich, Switzerland
[knorr, stormer] @ifi.unizh.ch

Abstract With the rise of global networks like the Internet the importance of workflow systems is growing. However, security questions in such environments often only address secure communication. Another important topic that is often ignored is the separation of duties which is an important part of a company's security policy to prevent fraud. This paper introduces a prototype that supports the graphical modeling and analysis of separation of duties in workflow environments. Security officers can use this tool to design and analyze the security rules associated with workflow specifications.

Keywords: Fraud Control, Separation of Duties, Workflow

1. INTRODUCTION

It is well known that many computer related criminal activities are performed by insiders [6]. One of the most prominent threats is fraud that is particularly difficult to detect in computerized environments such as workflow systems. Therefore it is of great importance to implement mechanisms to prevent such illegal activities. Separation of duties (SoD) has been identified by many authors as an efficient mechanism to prevent fraud within organizations [1, 2, 3, 20]. SoD guarantees that certain critical tasks can only be done by a collusion among individuals. It is in particular useful when applied to dynamic processes such as workflows. The physical and logical separation of tasks and of their performing subjects can improve the prevention of fraudulent activities.

SoD has been in use long before the computer era. One prominent example is the 'four-eye-principle' that is found in many environments such as health

care. Another example: Most military systems require (at least) two persons to launch a nuclear missile.

This paper introduces MASoD, a prototype supporting the graphical Modeling and Analysis of SoD-rules in workflow environments. Workflows are computer-understandable business processes whose modeling, administration, and execution is supported by a software package called workflow management system (WfMS). The effort associated with introducing a WfMS in an organization is tremendous. Especially (1) the implementation of the new system, integration and interfacing with existing systems, and (2) the identification, redesign and specification of the processes to be automated are time and resource consuming. Concerning the second point, we share the view of Huang and Atluri [9] who argue that security officers should utilize existing data when designing security policies. Therefore, MASoD is capable of importing existing workflow specifications, and provides users with organizational and process information to generate SoD-rules. Most workflow systems allow users to model workflows graphically. In continuation of this practice, SoD should be modeled in the same way. Also, in the past visual languages have proved to be user-friendly.

The paper has the following structure: Section 2 discusses the background topics of workflow management and SoD. A sample business process is introduced in Section 3. While Section 4 focuses on the graphical modeling of SoD-rules on top of existing workflow specifications, these rules are analyzed in Section 5. Section 6 concludes the paper with a discussion and an outlook.

2. BACKGROUND

2.1. WORKFLOW MANAGEMENT

Business processes represent an essential part of the commercial activity of a company. Especially for frequent processes, support for automation is an important topic. Workflow management is an emerging technology in the area of applied computer science dealing with this issue. A WfMS is a software system that supports the modeling, execution, and administration of business processes. Defined in four words, a workflow is a *computer-understandable business process*. Before a workflow can be executed it has to be described in a way the WfMS is able to understand. This description is called workflow specification and is done during the so-called *build time* of a WfMS. The most important part of a WfMS is the workflow engine which is responsible for the execution of the workflow during *run time* of the system when many instances of a workflow are created according to the workflow specification [4, 7, 15].

WfMSs are especially useful for electronic workflows. An electronic workflow is a workflow whose data items are stored in an electronic form. In this case the WfMS can manage and forward process-specific data items to the

subjects. The main elements of a workflow specification are: (1) tasks, (2) the control flow, and (3) subjects/roles — or more general the organizational structure. The basic building blocks of a workflow are its tasks whose temporal and logical order is given by the control flow. To describe a task, the activity and the corresponding subjects have to be specified. Subjects can be associated with persons, but also processes and computer programs such as the workflow engine are possible subjects. A subject executes a task by creating new and/or using already existing data items.

Prominent workflow specification languages are Petri Nets [16], State Charts [23] and the Workflow Process Definition Language proposed by the Workflow Management Coalition [24] — a non profit organization dealing with standardization in workflow systems.

In the last few years, the role concept has proven to be very successful in commercial settings. A role defines a group of human beings with a special knowledge or skill. Usually, a specific role is tied to each task in a workflow. Doing this, the administration of users is simplified since subjects can be added or removed without changing the workflow specification. In practice, an important application of the role concept is security. Access rights are not granted to single persons but to roles which simplifies and cheapens security administration. This is called role based access control [14, 17].

2.2. SEPARATION OF DUTIES

Gligor et al. [8] define SoD as “a policy to ensure that failures of omission or commission within an organization are caused only by collusion among individuals and, therefore, are riskier and less likely, and that chances of collusion are minimized by assigning individuals of different skills or divergent interests to separate tasks”. Clark and Wilson [5] stress the importance of SoD mechanisms in commercial settings. Within a business process context, SoD-rules express task dependencies and should be part of a company’s security policy. Only recently has the combination of workflow management and SoD found considerable interest in the research community [1, 2, 22].

In a workflow context, SoD has to be divided and extended into static and dynamic SoD. Static SOD enforces certain rules during build time of the workflow and is therefore applied to the workflow specification. In contrast, dynamic SoD is enforced during run time. For the remainder of this paper we will concentrate on dynamic SoD. Note that dynamic SoD is based on the history of a business process and can therefore only be enforced during run time of a WfMS.

3. SAMPLE PROCESS

The last section introduced several concepts on an abstract level. This section gives a sample business process that will be used for illustration purposes

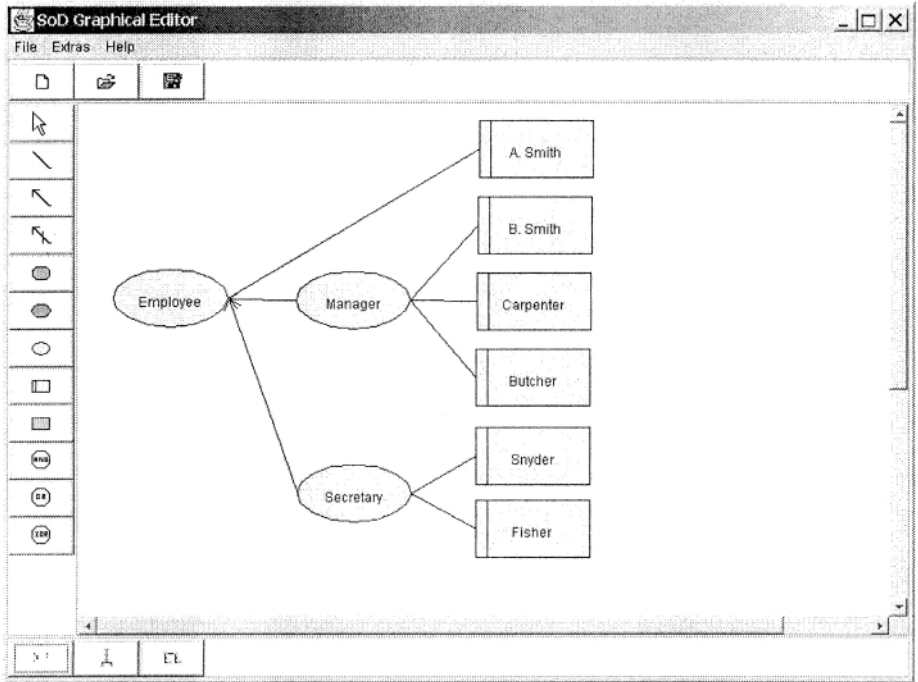


Figure 1 Sample organizational model. Six subjects and three roles are defined.

throughout the remainder of the paper. The business process deals with travel expense reimbursement. The workflow consists of four tasks: in a first task (*submit*), an employee applies for the reimbursement of his travel expenses by filling out an application form. Two managers have to approve this report (tasks *approve 1* and *approve 2*). Finally, based on the approval of the managers, a secretary will transfer the money to the employee's bank account (*pay*). Figure 3 gives a visualization. A discussion of the Notation will follow in Section 4.2. Note that the workflow engine creates an instance of this workflow for every travel of an employee.

Formally, the set of roles encompasses the manager, secretary, and employee role. Let the subjects of the company be Carpenter, Butcher, Snyder, Fisher, and the brothers A. Smith and B. Smith. All six subjects are employees, Carpenter, Butcher and B. Smith are managers, Fisher and Snyder are secretaries. Note that every manager (or secretary) is an employee, too. The partial order of roles builds up a so called role hierarchy. Figure 1 shows the subjects, roles, and the role hierarchy as modeled with MASoD.

Now we are ready to give SoD examples. According to the process definition different tasks are performed by different roles (the pay task by a secretary

and the approve 1 task by a manager). This corresponds to static SoD-rules. However, dynamic SoD is of greater interest for this paper. The following rules are reasonable for this specific business process:

- 1 A manager should not be allowed to approve his/her own travel expense claim.
- 2 B. Smith should not be allowed to approve the claim of his brother A. Smith.
- 3 A secretary should not be allowed to transfer the refund of his/her own travel expenses.
- 4 A manager should not be allowed to perform both approval tasks in the same workflow instance.

These examples clearly state the need for a formal model for SoD that takes into account the state of the underlying business process

4. MODELING OF SOD-RULES

Process and role definitions are modeled graphically in most workflow systems. Therefore, the modeling of SoD should also be done graphically. However, most of the existing SoD-languages are difficult to adapt for graphical notations. Besides, those languages are not intuitive and designed for security experts. We argue that the use of SoD depends on a simple language. Therefore, we introduce SSoDL (Simple SoD Language) and show how to graphically model SSoDL-rules in this section.

The modeling of SoD-rules requires the existence of organizational and process models. Therefore, MASoD (our prototype) supports the modeling of three different components:

- roles, subjects, and role hierarchy
- process
- SoD-rules

Note that the models should be created in the order indicated in the listing.

Section 4.1 will briefly discuss how MASoD handles organizational and process models. However, the major focus is on the graphical modeling of SoD-rules in Section 4.2.

For the comfort of the reader, Figure 2 gives a summary of all symbols used for the three models supported by MASoD. All the symbols will be explained in the following subsections.


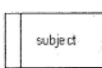





							
Role Model		●			●		
Process Model			●	●	●	●	
SoD Model	●	●	●				●

Figure 2 Symbols used in MASoD. Each row of the table represents a modeling component. Each column is one or more symbols. A bullet indicates that the symbol(s) is/are used in the modeling component.

4.1. GRAPHICAL MODELING OF ROLES AND PROCESSES

MASoD is designed to import existing process definitions. If the tool is used on a stand-alone basis, a workflow can also be specified graphically as an EPC (Event-driven Process Chain)— a model that was introduced by Scheer [21] and is frequently used in Europe, especially in German speaking countries. EPCs are an important part of a more complex product family called ARIS that allows for modeling all major components of a company.

We now discuss how the organizational structure is modeled using the MASoD editor. Figure 1 shows the organizational definition corresponding to the sample discussed in Section 3. Roles are represented as ellipses, subjects as rectangles, and the role membership as lines connecting subjects with roles. The role hierarchy is illustrated as roles connected with arrows. Six subjects and three roles are defined. Carpenter, for instance, is assigned the manager role. Because of the role hierarchy, Carpenter is an employee, too.

When the organizational model has been designed, the process modeling can commence. Note that we use EPC terminology. Every workflow is identified with one EPC. An EPC consists of an alternating chain of events and functions. It starts and ends with an event. A function is used to model a task. Only roles that have previously defined in the organizational model can be applied to functions, not subjects. Graphically, a function is represented as a rounded box, an event as a sexangle. One role has to be tied to each function which is visualized through a connector. Arrows represent the control flow. Branchings are modeled using AND, OR, and XOR nodes. Figure 3 gives the EPC according to our example. Scheer [21] gives a more comprehensive introduction to EPCs.

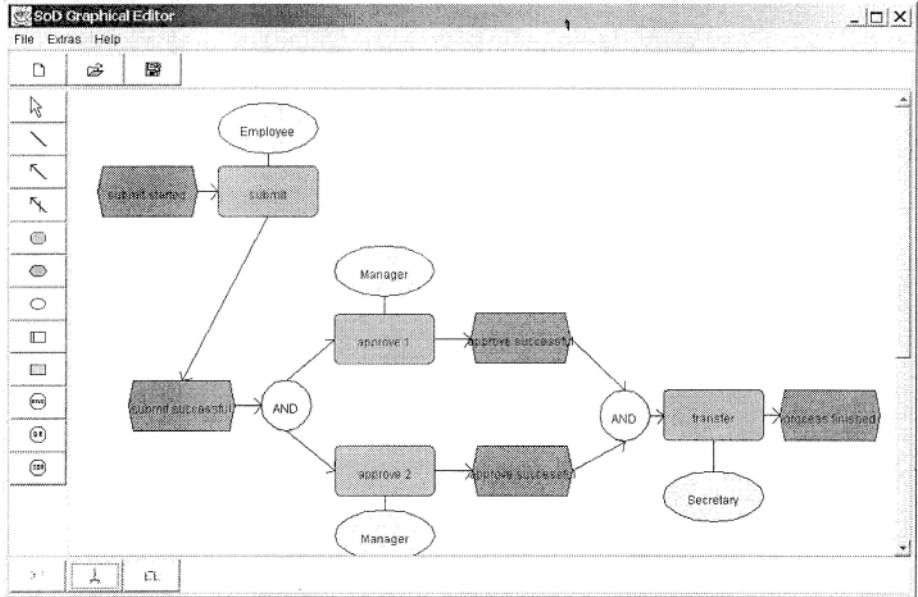


Figure 3 Sample process model with four tasks.

4.2. GRAPHICAL SOD MODELING

The basis of our graphical model is SSoDL (Simple SoD Language) which describes SoD-rules. These rules have the following form:

$$(s_1, t_1) \rightarrow (s_2, t_2) \tag{1}$$

or

$$(s_1, t_1) \not\rightarrow (s_2, t_2) \tag{2}$$

Rule (1) represents a **delegation of duties**. If subject s_1 has performed task t_1 , then subject s_2 has to perform task t_2 . Rule (2) represents a **separation of duties**, if subject s_1 has performed task t_1 , then subject s_2 must not perform task t_2 . An example from Section 3: (Butcher, approve 1) $\not\rightarrow$ (Butcher, approve 2).

Note that the legitimacy of a subject to perform a task is given by the role membership of the subject, i.e. only subjects that are member of the role associated with task t_1 are allowed to perform task t_1 . In an EPC, for each task there is exactly one role. Note also that $s_1 = s_2$ is possible, but that t_1 must precede t_2 . The partial order of the tasks is given by the underlying EPC.

The rule that a manager should not do both approval tasks would result in 12 SoD-rules if rules could only be defined on a subject/task level, which would be quite unsatisfactory in environments with a large number of subjects.

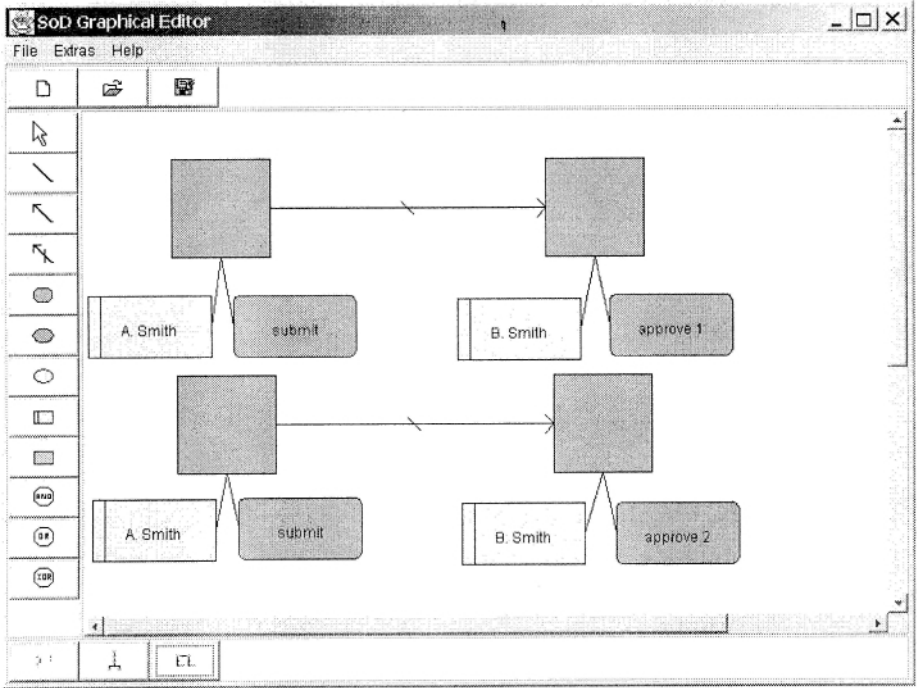


Figure 4 Graphical modeling of the brother example.

Therefore, we introduce the ?-notation: For s_1 and s_2 in (1) and (2), a ? can be inserted, $(?, t_1) \rightarrow (s_2, t_2)$ means that rules of type (1) are generated in the MASoD analyzer (cf. Section 5) for all subjects that are member of the role associated with task t_1 .

The ?-notation can also be used on both sides of (1) and (2). Example: $(?, t_1) \not\rightarrow (s_2, t_2)$. In this case, subject/task rules of type (2) are generated for all subjects who can perform task t_1 and t_2 . Note that this practice requires a role hierarchy or the membership of subjects to multiple roles. An example from Section 3: The two rules $(?, submit) \not\rightarrow (s_2, approve 1)$ and $(?, submit) \not\rightarrow (s_2, approve 2)$ would prevent managers from approving their own travel claims.

We will now discuss how to model these rules graphically. The subject/task tuples are illustrated as rectangles (subjects) and rounded boxes (tasks or functions in EPC terminology). Note that MASoD only allows for subjects and tasks which have previously been defined in the organizational and process model. To represent subject/task tuples, subjects and tasks are connected to a square. Separation and delegation arrows (\rightarrow and $\not\rightarrow$) are used between two squares. The graphical modeling is illustrated using the example rules 2 and 4 from the sample business process. The first rule discussed is the brother example from

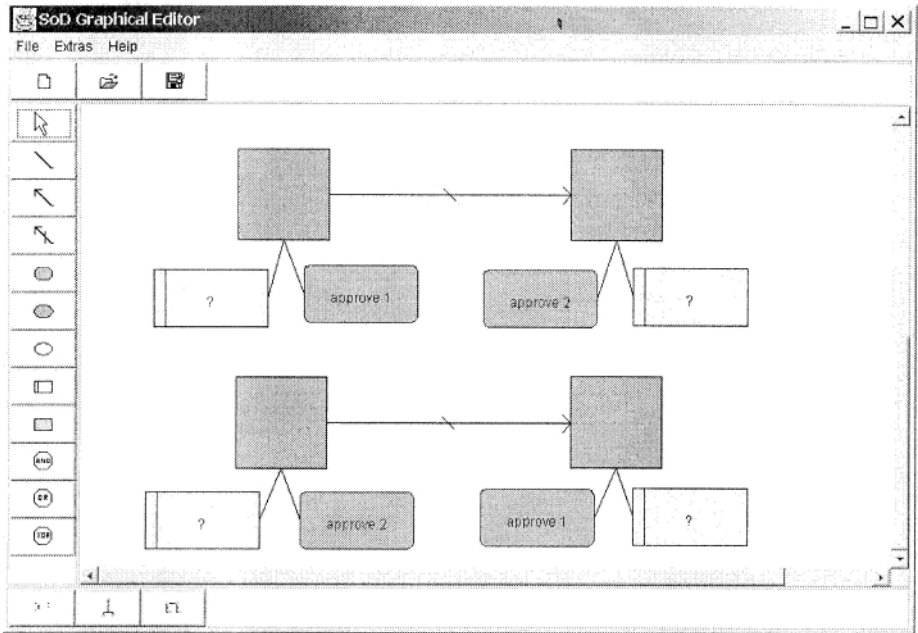


Figure 5 Second graphical SoD example with ?-notation.

Section 3. If A. Smith has done the submit task, then B. Smith is not allowed to do one of the approve tasks. Using the SSoDL syntax, this rule can be expressed as:

(A. Smith, submit) $\not\leftrightarrow$ (B. Smith, approve 1)
 (A. Smith, submit) $\not\leftrightarrow$ (B. Smith, approve 2)

A graphical model of this rule in the MASoD editor is shown in Figure 4. Note the one-to-one relation between the SSoDL rules and the graphical notation. In the fourth SoD example rule in Section 3, a manager is not allowed to do both approve tasks. This can be expressed using the ?-notation:

(?, approve 1) $\not\leftrightarrow$ (?, approve 2)
 (?, approve 2) $\not\leftrightarrow$ (?, approve 1)

Figure 5 shows the graphical modeling of this example. Note that the analyzer is creating the following six rules by substituting the ?:

(Butcher, approve 1) $\not\leftrightarrow$ (Butcher, approve 2)
 (B. ~Smith, approve 1) $\not\leftrightarrow$ (B. ~Smith, approve 2)
 (Carpenter, approve 1) $\not\leftrightarrow$ (Carpenter, approve 2)
 (Butcher, approve 2) $\not\leftrightarrow$ (Butcher, approve 1)

(B.~Smith, approve 2) $\not\rightarrow$ (B.~Smith, approve 1)
 (Carpenter, approve 2) $\not\rightarrow$ (Carpenter, approve 1)

5. SOD ANALYSIS

Section 4 introduced SSoDL, a language for SoD, and its graphical modeling. On the basis of this, we will now show how to analyze the resulting SoD-rules. This section only gives an overview. The detailed description of the SoD analysis including a comprehensive description of the Prolog program and the analysis technique can be found in [12].

The rules for each process are contained in a so-called *Rule Base* (RB). The RB will only consist of rules as shown in (2). That means that the ?-notation has to be removed (cf. Subsection 4.2) and that rules of type (1) have to be transformed to rules of type (2) according to the following equivalence:

$$(s_1, t_1) \rightarrow (s_2, t_2) \iff [(s_1, t_1) \not\rightarrow (x, t_2) \quad \forall x \in S \setminus \{s_2\}]$$

where S is the set of all subjects. For a discussion of the complexity of this transformation see [12].

The analysis of the SoD-rules makes use of logical programming — to be precise Prolog. As a preparation, the process model (EPC) is transformed into a Petri net [10, 18]. The transformation step is well understood [13, 19]. The Petri net representation can be seen as a formalized version of the EPC. Now, the following steps are necessary:

- Transformation of the organizational structure into facts of the logical program
- Transformation of the Petri net into facts of the logical program
- Transformation of the SSoDL-rules into facts of the logical program

After the transformations, MASoD checks if the rules are sound, i.e. if the rules obey to the organizational and process structure. Example: the rule

(A. Smith, pay) \rightarrow (Butcher, submit)

is not sound since A. Smith is not a secretary and the submit task precedes the pay task. Given a sound rule-base, the program generates all valid execution chains of the workflow. An execution chain consists of subject/task tuples that represent the firing order of the underlying Petri net. If there is no contradiction to the SoD rule-base, the execution chain is called valid. Examples from our sample: Let L_1 , L_2 , L_3 be three chains with

```

L1 = [(Fisher, submit), (A. Smith, approve 1),
      (Butcher, approve 2), (Snyder, pay)]
L2 = [(Fisher, submit), (Butcher, approve 1),
      (Butcher, approve 2), (Fisher, pay)]
L3 = [(Fisher, submit), (Carpenter, approve 1),
      (Butcher, approve 2), (Snyder, pay)]

```

L1 is no execution chain since A. Smith is no legitimate subject for one of the approve tasks (a manager is needed). L2 is an execution chain but not SoD valid since Fisher transfers his/her own travel expenses and Butcher does both approval tasks. L3 is SoD valid.

The Prolog program generates 28 valid execution chains in our example:

1. [(A.Smith, submit), (Carpenter, approve 1),
(Butcher, approve 2), (Fisher, pay)]
2. [(A.Smith, submit) , (Carpenter, approve 1),
(Butcher, approve 2), (Snyder, pay)]
- ...
28. [(Snyder, submit), (Carpenter, approve 1),
(Butcher, approve 2), (Fisher, pay)]

A further analysis of these 28 valid execution chains yields the following:

- For every employee in the example, the travel expense workflow can be finished. No workflow has to be canceled due to too restrictive SoD-rules. Although this statement may seem obvious for the example, in a more complex setting it is not.
- Every workflow requires four different persons for its execution. Without the SoD-rules two persons would be sufficient (e.g. first three tasks by a manager, last one by a secretary).
- A further analysis of the valid execution chains can be used to do ‘workload management’. In the example, the managers Carpenter and Butcher have more work because of the ‘brother rule’ (cf. Table 1).
- The analysis shows that the rule base of the example contains no contradicting rules.

Technically, MASoD generates three files: the organizational, Petri Net and SoD Prolog facts. Then, the Prolog program is called that reads the files, processes them as input, and generates an output file. The data from the output file is read into MASoD and is graphically displayed.

$N(s, t)$	A.Smith	B.Smith	Carpenter	Butcher	Snyder	Fisher
submit	4	4	4	4	4	4
approve 1	0	8	10	10	0	0
approve 2	0	8	10	10	0	0
pay	0	0	0	0	14	14

Table 1 Statistical analysis of the sample workflow. $N(s, t)$ is the number of valid execution chains in which subject s performs task t

6. CONCLUSION AND OUTLOOK

Current surveys [6] show that fraudulent activities performed by insiders are a tremendous threat for the commercial success of a company. Therefore, appropriate counter-measures such as SoD have to be found and implemented.

The MASoD prototype supports the modeling of the organizational structure, processes, and SoD-rules. Our SoD-modeling provides a graphical, user-friendly approach to define SoD-rules. Furthermore, these rules can be checked for compliance with workflow specifications to prevent flaws during run time of the system. We tried to keep the complexity of the rule language (SSoDL) on a comprehensible level in order to provide a user-friendly graphical interface. Therefore, the expressiveness of SSoDL is smaller in comparison to other SoD-languages [1, 2, 3]. Nevertheless, our model catches most of the real-life rules and is intuitive to use.

Future research will encompass the following issues:

- Extension of SSoDL (AND, XOR, and OR connectors)
- Empirical test of the expressiveness and applicability of MASoD in a real-life setting
- Better support of import and export facilities to and from other workflow specifications such as the Workflow Process Definition Language proposed by the Workflow Management Coalition.
- The modeling and analysis of SoD rules is only a small part of a larger picture. The enforcement of SoD during run time, i.e. during the execution of workflow instances, is an important issue. Knorr [11] introduced an architecture for a workflow system based on Petri nets. We plan to use this architecture and enforce SoD based on SSoDL.

References

- [1] Gail-Joon Ahn and Ravi Sandhu. The RSL99 Language for Role-based Separation of Duty Constraints. In *Proceedings of the Fourth ACM Workshop on Role-Based Access Control*, Fairfax, VA, October 28-29 1999.
- [2] Elisa Bertino, Elena Ferrari, and Vijay Atluri. The Specification and Enforcement of Authorization Constraints in Workflow Management Systems. *ACM Transactions on Information and System Security*, 2(1):65–104, Feb. 1999.
- [3] Christoph J. Bussler. Policy Resolution in Workflow Management Systems. *Digital Technical Journal*, 6(4), 1995.
- [4] Andrzej Cichocki, Abdelsalam Helal, Marek Rusinkiewicz, and Darrell Woelk. *Workflow and Process Automation — Concepts and Technology*. Kluwer Academic, 1998.
- [5] David R. Clark and David R. Wilson. A Comparison of Commercial and Military Computer Security Policies. In *Proceedings of the 1987 IEEE Symposium Security and Privacy*, pages 184–194, Oakland, CA, 1987.
- [6] Computer Security Institute. Issues and Trends — CSI/FBI Computer Crime and Security Survey. [http : //www . gocsi . com/summary . html](http://www.gocsi.com/summary.html) , 1999.
- [7] Dimitrios Georgakopoulos, Mark Hornick, and Amith Sheth. An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3:119–153, 1995.
- [8] Virgil D. Gligor, Serban I. Gavilla, and David Ferraiolo. On the Formal Definition of Separation-of-Duty Policies and their Composition. In *Proceedings of the 1998 IEEE Symposium on Security and Privacy*, Oakland, CA, 1998.
- [9] Wei-Kuang Huang and Vijayalakshmi Atluri. SecureFlow: A Secure Web-enabled Workflow Management System. In *Proceedings of the Fourth ACM Workshop on Role-Based Access Control*, pages 83–94, Fairfax, VA, October 28-29 1999.
- [10] Kurt Jensen. *Coloured Petri Nets — Basic Concepts, Analysis Methods and Practical Use, Volume 1*. EATCS Monographs on Theoretical Computer Science. Springer, 1992.
- [11] Konstantin Knorr. WWW Workflows based on Petri Nets. In *Proceedings of the Ninth International Conference on Information Systems Development*, Kristiansand, Norway, August 2000.
- [12] Konstantin Knorr and Harald Weidner. Analyzing Separation of Duties in Petri Net Workflows. In *Proceedings of the First International Workshop on Mathematical Methods, Models and Architectures for Computer Networks Security*, St. Petersburg, May 21-23 2001.

- [13] Peter Langner, Christoph Schneider, and Joachim Wehler. Prozessmodellierung mit ereignisgesteuerten Prozessketten (EPKs) und Petri-Netzen. *Wirtschaftsinformatik*, 39(5):479–489, 1997.
- [14] L. G. Lawrence. The Role of Roles. *Computers & Security*, (12): 15–21, 1993.
- [15] F. Leymann and W. Altenhuber. Managing Business Processes as an Information Resource. *IBM Systems Journal*, 33(2):326–348, 1994.
- [16] Andreas Oberweis. *Modellierung und Ausführung von Workflows mit Petri-Netzen*. Teubner-Reihe Wirtschaftsinformatik. B.G. Teubner, 1996.
- [17] *5th ACM Workshop on Role-Based Access Control*, Berlin, July 2000.
- [18] Wolfgang Reisig. *Petri Nets—An Introduction*. EATCS Monographs on Theoretical Computer Science. Springer, 1985.
- [19] P. Rittgen. Paving the Road to Business Process Automation. In Martin Bichler and Harald Mahrer, editors, *Proceedings of the 8th European Conference on Information Systems (ECIS)*, volume 1, pages 313-319, Vienna, Jul. 2000.
- [20] Ravi Sandhu. Separation of Duties in Computerized Information Systems. In *Proceedings of the IFIP WG11.3 Workshop on Database Security*, Halifax, UK, September 18-21 1990.
- [21] August-Wilhelm Scheer. *Wirtschaftsinformatik. Studienausgabe. Referenzmodelle für industrielle Geschäftsprozesse*. Springer, 1998.
- [22] Henrik Stormer, Konstantin Knorr, and Jan Eloff. A Model for Security in Agent-based Workflows. *Informatik • Informatique*, 6:24–29, Dec. 2000.
- [23] Dirk Wodtke and Gerhard Weikum. A Formal Foundation for Distributed Workflow Execution Based on State Charts. In *Proceedings of the international Conference on Database Theory*, Greece, 1997.
- [24] Workflow Management Coalition. *Interface 1: Process Definition Definition Interchange – Process Model*. Workflow Management Coalition, 1998. Document Number TC-1016-P.

Group Security Association (GSA) Management in IP Multicast

Thomas Hardjono, Mark Baugher and Hugh Harney
Nortel Networks, Cisco Systems, Sparta Corp.

Key words: Group security, multicast security, key management, Security Associations, IPSEC, IKE, IETF.

Abstract: This work describes the Group Security Association (GSA) Management model and protocol as developed in the Secure Multicast Group (SMuG) in the IETF. The background reasoning from the Internet Key Exchange (IKE) protocol perspective is explained, together with the notion of Security Associations (SA) in the unicast case. This serves as a basis for a requirements for Group SA for multicast. Finally, the definition and construction of a GSA is described.

1. INTRODUCTION

There is significant interest in the networking industry and content delivery network industry to use IP multicast a vehicle for data delivery to a large audience. One major hindrance to the successful deployment of IP multicast and other group-oriented communication protocols has been the lack of security for both the content and the content-delivery infrastructure. In particular, there has been increasing demand for secure solutions for the 1-to-Many type of group communications, as exemplified by the interest of the cable television sector in using the Internet for content distribution and by the recent emergence of the single-source paradigm in IP multicasting.

To this end, the IETF designated in mid-1998 the creation of the Secure Multicast Group (SMuG) under the umbrella of the Internet Research Task Force (IRTF) to research and develop protocols for multicast security. The approach adopted was to address the issues surrounding multicast security in three related problem-areas. These three problems-areas correspond to

issues relating to the transformation of multicast data, group key management and group security policy management.

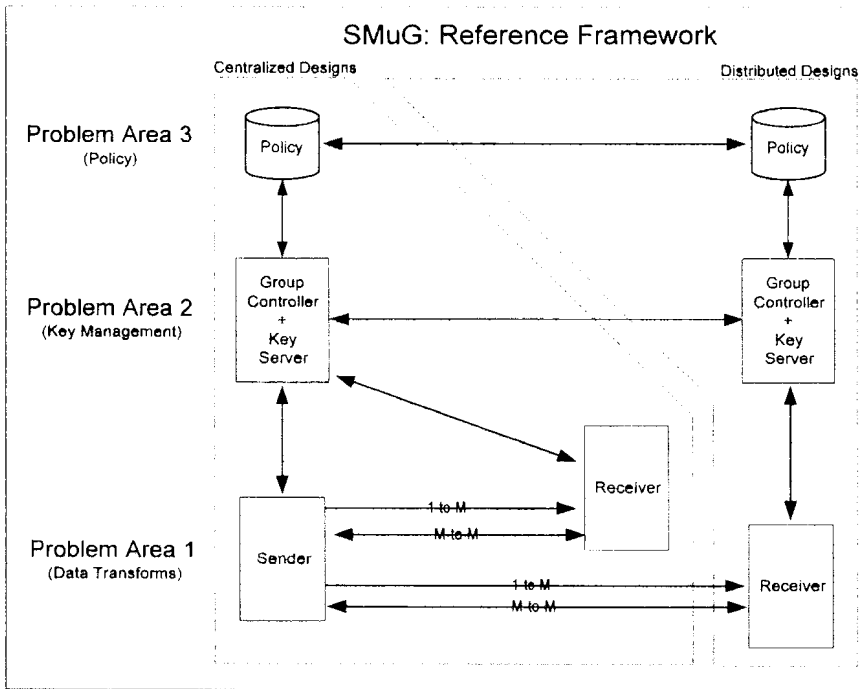


Figure 1. Secure Multicast Group (SMuG) Framework [HCBD00]

A crucial component of group key management is that of the management of Security Associations (SA). In the unicast world, SA management for pair-wise communications has been addressed by the ISAKMP framework [RFC2408], and has been embodied in the Internet Key Exchange (IKE) protocol [RFC2409]. As understood by IKE, ISAKMP (and more broadly, the IETF), the term “key management” incorporates the wider aspects of keying material, including cryptographic keys, key identities, and other parameters that support the establishment of common (symmetric) keys and both ends of a unicast connection.

In the context of IP multicast, and the wider field of group communications, the SA management model underlying ISAKMP/IKE is insufficient due to the fact that a group has many members (Senders and Receivers). In particular, the notion of SA negotiations is not applicable since multi-party negotiations of SA parameters in multicast is impractical and resource consuming for many multicast applications.

The current paper describes work that has been on-going in SMuG since mid-1998 [SMuG00]. Section 2 briefly touches on the Framework developed in SMuG [HCBD00]. Section 3 describes the requirements of key management in unicast sense, and extends it to the multicast situation. Section 4 defines the notion of the Group SA and provides some examples, whilst Section 5 concludes the paper with some remarks and future directions.

2. BACKGROUND: SMUG FRAMEWORK

The Secure IP Multicast Framework and Building Blocks document [HCBD00] describes a number of entities, which participate in the creation, maintenance, and removal of secure multicast groups (Figure 1). Those that are immediately of concern for group key management and membership management are as follows:

- *Group Controller and Key Server (GCKS):*
The GCKS entity embodies both the physical entity and functions of the group controller and the key server. Although two families of functions can be distinguished, namely membership management and group key management, for simplicity both families of functions will be provided by a single physical entity. For any given multicast group, a “Main” or “Root” GCKS must be identified using methods and mechanisms related to a group's initial definition/configuration.
- *Member (Receiver and Sender):*
The member is the group member, defined for a particular instance of group communications. The member entity can exist at different layers (eg. user, host, process) and thus must be defined across the group consistently and is best expressed through their corresponding certificates.

Although not directly addressed in the current document, another entity that is involved in group key management is the Remote GCKS. This is expressed in the context of Distributed Designs in Figure 1. The Remote GCKS expresses the scalability and inter-domain requirements of group key management. A Remote GCKS is identical in functionality to the GCKS. However, in terms of authorization level to perform the management of group keys, a GCKS may possess a different relationship to another GCKS within the same management regime. Examples include a peer relationship

among a set of GCKS, and a hierarchical relationship where a Root GCKS is defined and the other GCKS are subordinates of the Root.

3. GROUP KEY MANAGEMENT: REQUIREMENTS AND PROPERTIES

The requirements discussed in this section are for the most part not original but come from a variety of sources. The Internet key management literature is one source. Group key management must operate over packet internetworks, particularly IP multicast internets. Thus group key management has at least some of the properties of Internet key management. Indeed, the very notion of "key management," as distinct from "key exchange," is taken from work done on IPsec. Thus, the Internet key management requirements presented in this paper are gleaned from prior work done on IP security, key management for packet networks, and authenticated key exchange [RFC2409, RFC2412, RFC2408, RFC2407, RFC2522, Kraw96, SDNS88, DVW92]. Our second source of requirements is taken from previous work on multicast security [RFC2627, CP99, HH99a, HCD99].

Group key management requires additional properties beyond those found in the Internet key management work done to date. Group keying material, for example, are not negotiated but sent to and shared by groups of members, which must agree to common policies that are not negotiated [CP99, HH99a]. Furthermore, the key exchange/distribution architecture is not only peer-to-peer but also operates between key server and key client [HCBD00]. The common and distinct properties of Internet key management and group key management are the subject of this section.

Section 3.2 discusses group key management. Section 3.1 is an overview of internet key management and its applicability to group key management. In both sections we consider the needed properties of a group key management protocol.

3.1 Internet Key Management and Key Determination

The "authenticated key exchange" (AKE) notion is basic to inter-net key management and key determination protocols, which seek to thwart attacks that may occur on an unsecured network. The types of attacks include man-in-the-middle, connection-hijacking, and reflection/replay attacks, many of which can be combated by mechanisms such as "direct authentication", which integrate authentication into the key exchange, as described in the STS protocol [DVW92]. Messages that are exchanged as part of a "run"

should be chained with authenticable information, including random data that is contributed by each party in a two-party key exchange. This technique helps ensure that messages received by a peer match what the other peer sent. Work has been done, moreover, to formally prove AKE properties based upon the matching of messages sent and received by peers in the exchange [BR93]. When session keys are used to protect exchanges that determine other session keys, "perfect forward secrecy" (PFS) can ensure that *"...disclosure of long-term secret keying material does not compromise the secrecy of the exchanged keys from earlier runs"* - so long as authentication is linked to the key exchange [DVW92]. The PFS requirement, however, entails the performance penalty of a Diffie-Hellman exchange, which may not be appropriate for all applications.

The notion of a "selectable level of security" is also basic to key management on internetworks, which are composed of diverse communications networks and host computers. In this environment, some applications may trade-off better security for reduced communications and computing costs. The security choices depend upon application need as well as the capabilities of the hosts and network devices. In order to support heterogeneous network and host devices, Internet key management supports multiple types of exchanges that can be composed in various ways; some exchanges may support identity protection and provide PFS, for example, while others may not [Kraw96]. To accommodate diversity, a versatile approach supports a variety of transforms and Diffie-Hellman groups, all of which can be negotiated among communicating entities [RFC2412, RFC2409]. Internet key management, moreover, supports a "forward migration path" in the protocol so that new algorithms can be introduced, as older methods need to be replaced [RFC2409, RFC2412, RFC2408, Kraw96].

In fact, the key establishment procedure itself may need to be replaced over time, and the Internet Security Architecture has a key management framework, the Internet Security Association and Key Management Protocol (ISAKMP), which defines an abstract set of exchanges, organized by modes and phases to provide a selectable level of protection [RFC2408, Kraw96]. To provide a versatile solution for internet key management, ISAKMP permits alternative authentication mechanisms in its exchanges and is parameterized by a domain of interpretation (DOI) in which specific key determination mechanisms are defined through the specification of the name space, policy, specific payloads and, optionally, new exchanges. In this way, ISAKMP is designed to be extended for alternative uses and to allow a forward migration of key exchange protocols and cryptographic transforms. Although the flexibility of their approach may arguably result in more complexity, which may in turn lead to weaker security, the ISAKMP authors

recommend the use of ISAKMP as a single key management framework for new uses such as group key management, as well as transport and application key management [RFC2408]. New uses can be realized through the specification of a DOI.



Figure 2. A Security Association (SA) between two entities

ISAKMP achieves its versatility by being more abstract than a key determination protocol since it manages security associations (SA) and not just keys. The SA abstraction [RFC2408, RFC2401, RFC2522, SDNS88] encapsulates keys and information about keys, such as key lifetimes and cryptographic policies, so as to allow all significant aspects of the security to be modified to the needs of the application and environment. In the current Internet Security Architecture, however, SA management is peer to peer as depicted in the Figure 2.

The SA is defined to be simplex in the Internet Security Architecture [RFC2401] and is identified by a Security Parameter Index (SPI) [RFC2401, RFC2522]. SAs are established according to local policy [RFC2401, SDNS88] using exchanges that are designed to protect against basic key establishment attacks, such as man in the middle, connection hijacking, replay/reflection, and denial of service [RFC2408]. Although the first three types of attacks are the subject of authenticated key exchange mechanisms, protection against the denial-of-service attack uses a pairwise cookie mechanism [RFC2522] between peer entities, which appears used in the ISAKMP header for all exchanges [RFC2408, RFC2409].

Since we assume that group key management must operate across diverse internetworks, particularly IP multicast networks, then at least some of the properties of Internet key management are required for group key management. These properties, broadly stated, are summarized in the points below:

1. Protection against man-in-the-middle, connection-hijacking, replay/reflection, and denial-of-service attacks.

2. Selectable level of security protection in key establishment, such as alternative transforms, optional PFS and identity protection to support heterogeneous internet applications and computers.
3. Alternative authentication mechanisms such as shared key, PKI, and public key to support diverse trust models.
4. Forward migration path for new security mechanisms such as new cryptographic transforms and even new exchanges.
5. A single key management framework to support the establishment of Security Associations according to the local policies of internet host and intermediate systems.

We assume that these properties should be properties of group key management as well. As discussed next, group key management has additional needs beyond the five points summarized above.

3.2 Group Key Management

From the previous section, it is clear that many of the requirements and design features of Internet key management are needed by group key management. In fact, many of the payloads, exchanges, and transforms found in ISAKMP and IKE may be suitable for group key management: Many group key management protocols and algorithms, moreover, such as GKMP, LKH, OFT, GSAKMP, NARK and MARKS assume a unique key for a member, which is established using point-to-point procedures with a key server [RFC2093, RFC2094, RFC2627, BMS99, HH99b, BF99, Bris99]. For the purposes of authenticating a potential group member and initializing it with keys, group-keying material must be “pulled” by an individual client from the server. Group members whose computers are off-line during key updates also must pull keying material to be re-initialized (or to request re-initialization by the GCKS) in a secure, probably point-to-point protocol. Use of IKE unchanged (with the IPsec DOI), however, is out of the question owing to the need to support key distribution in addition to exchange (i.e., an external key is given to the member by the GCKS), the need for policy distribution rather than policy negotiation, and the use of multicast communications to push key updates to promulgate key changes needed to refresh keys that reach the end of their cryptographic lifetime and to replace keys resulting from changes in group membership. Several algorithms have been proposed to efficiently accomplish group re-key and maintenance [RFC2627, BMS99, HH99b, Bris99]. A versatile group key management building block will support a variety of alternative algorithms to offer a forward migration path when new algorithms are developed or flaws in existing algorithms are uncovered.

The use of a multicast service to “push” key updates and other control messages from the GCKS to members relieves the GCKS of the burden of contacting each member individually to change the key or the configuration of the group [CP99, HH99a, RFC2627, BMS99, HH99b]. In this way, group key management can scale to very large numbers of members. This ability to deploy multicast itself for group key management is attractive for a variety of applications. This property may be superfluous for pure pay-per-view sessions where the member is keyed once and never again for duration of the session. But for subscription sessions or sessions where keys must be changed, a good multicast application design principles will protect the GCKS from being the target of periodic, and possibly synchronized, requests from large numbers of members attempting to pull keys.

Unlike large-scale subscription groups, short-lived, dynamic groups, which are characterized by relatively small numbers of members, may need group key management to minimize the time it takes to create and add members to a group. Thus, group key management must be able to efficiently maintain very large, secure groups, to support large numbers of members, while not precluding fast initialization, maintenance, and destruction for smaller groups that engage in impromptu group communications [CP99, RFC2627, HH99b]. The need to support a range of performance and scalability needs for diverse applications is very much a goal of Internet key management that is shared by group key management.

It is clear, however, that the security associations for group key management are more complex, or at least more numerous, than for unicast key management. Whereas the latter establishes a key management SA to protect application SAs (where a minimum of two are needed to key an Internet application process), group key management requires at least three: There is a “pull” SA between the group member and the GCKS, a “push” SA between the GCKS and all the group members, and an SA to protect application data from sender-members to receiver-members. In fact, each sender to the group may use a unique key for their data and use a separate SA: there may be more SAs than there are group senders.

Group key management, therefore, uses a different set of abstractions than ISAKMP and IKE. The abstractions used, however, may be built from the ISAKMP abstractions: in our approach, the Group Security Association (GSA) includes the attributes of the Internet Security Architecture SA, which is succinctly defined as the encapsulation of keys and policies [RFC2409] as follows:

- a SA has selectors, such as source and destination transport addresses.

- a SA has properties, such as an security parameter index (SPI) or cookie pair, and identities.
- a SA has cryptographic policy, such as the algorithms, modes, key lifetimes, and key lengths used for authentication or confidentiality.
- a SA has keys, such as authentication, encryption and signing keys.

As is discussed in the next section, a GSA contains the SA attributes plus some additional ones:

- a GSA has group policy attributes, such as the kind of signed credential needed for group membership and whether the group will be given new keys when a member is added (called "backward re-key" below) or whether group members will be given new keys when a member is removed from the group ("backward re-key").
- a GSA has SAs as attributes.

The final point, a GSA includes multiple SAs, is graphically depicted in Figure 3 and discussed more fully in the next section.

The following list summarizes the desired properties of Internet group key management:

1. The five properties of Internet key management as described in the previous section.
2. Support for the IRTF Secure Multicast (SMuG) Reference Framework, having a GCKS that controls access to the group of sending and receiving members according to the group policy it distributes
3. Support for IP multicast applications where there may be one or more senders to the group who may each have a unique SA to the group or who may each share a common SA to the group.
4. Support for both receiver-initiated "pull" of policy and keying material in addition to server-initiated "push" using a variety of re-key algorithms.
5. Selectable level of performance for group key management, which permits tradeoffs in startup latency, re-initialization complexity, message overhead, join latency, leave latency, and other security-related performance such as transforms.

Group key management requires a protocol with the five properties listed above. The protocol must be capable of establishing security relationships that are not just peer-to-peer but also between GCKS and a group of members (e.g., for re-key) and among sending and receiving members (e.g.,

for data protection). This section suggested that these relationships might be built upon group security associations, which in turn build upon the security association concept of IPsec and ISAKMP/IKE, as described in the next section.

4. GROUP SECURITY ASSOCIATION: REASONING AND DEFINITION

4.1 Structure of a GSA: Reasoning

There are three categories of SAs aggregated into a GSA. We choose this structure to better realize a GSA in our key management environment, the SMUG Reference Framework [HCBD00]. There is a need to maintain SAs between a Key Server and a group member (either a sender, a receiver or both) and among members. In the SMUG Reference Framework, the Key Server is called the “GCKS”, which is charged with access control to the group keys, with policy distribution to client members or prospective members, and with group key dissemination to sender and receiver client members. This structure is common in many group key management environments [HH99a,HH99b, CP99, RFC2627, BMS99, Bris99]. There are two SAs established between the GCKS and the members (Category-1, or SA1 and Category-2, or SA2), and there is an SA established among the sending and receiving members (Category-3 or SA3) as shown in Figure 3. The term SA1, SA2 and SA3 is used to simplify the following discussion.

The first category of SA (namely SA1 in Figure 3) is initiated by the member to pull GSA information from the GCKS; this is how the member requests to join the secure group or has its GSA keys re-initialized after being disconnected from the group (e.g., when its host computer has been turned off during re-key operations as described below). The GSA information pulled down from the GCKS include the SA, keys and policy used to secure the data transmission between sending and receiving members; this is SA3 in Figure 3. Note that SA3 is a category of SA, and this implies that there may be multiple SAs established between member senders and member receivers - at least as an option. There may exist, for example, a single SA of category SA3 in which all senders share common keys and associated information. Alternatively, there may be one or more SAs of category SA3 that are unique to the particular sender. An SA3 security association may be re-established or have its keys modified through re-key operations, which occur over an SA of category SA2. Keys are pushed through an SA of category SA2 to support subscription groups.

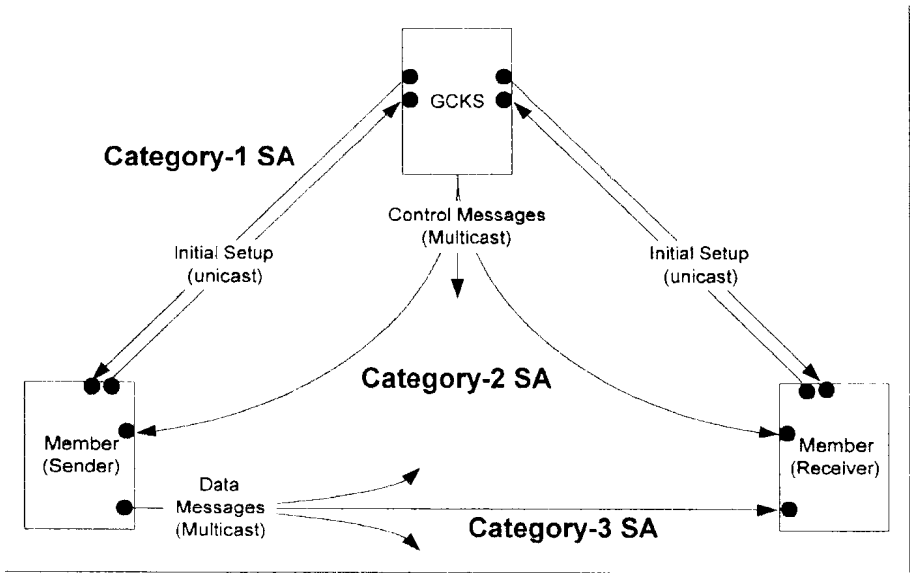


Figure 3. GSA Structure

Thus, the aim is to use SA1 to initially securely download SA2 and SA3 from the GCKS to the members. SA2 is then used for control messages, while SA3 for data messages. Included in the set of control messages is the update or replacement of SA3. Thus, we say that SA2 is used to “update” SA3, since it is anticipated that there will be far fewer use of SA2 compared to SA3 (e.g. SA3 for voluminous streaming media data). Naturally, the cryptographic policy for SA2 must specify strengths equal or stronger than SA3. Two options (at least) are available for “updating” the SA2 in turn. The SA2 can be updated through SA1 again (unicast) or the “old” SA2 can be used to update to a “new” SA2. We have left this as an implementation option, since the definition of a GSA must cater for a wide variety of applications.

Note that for applications where key updates occur within the data stream (protected using SA3), the GSA definition requires that SA2 be declared as “null” (which is different from saying it is non-existent). This is also true for group key management schemes that rely solely on point-to-point. Most others combine unicast exchanges for initialization with multicast distribution for re-key. In some cases, such as in a pure pay-per-session (PPV) application, all of the SA information needed for the session may be distributed at the time of registration or selection of a session (i.e. over an SA1); re-key and re-initialization may not be necessary, so there is no SA2. For subscription groups where keying material is changed as membership

changes, an SA2 is needed to re-initialize an SA3. Hence, in summary, the GSA concept sees the three Categories of SAs as being inseparable.

4.2 Definition of GSA

A GSA is defined to include an aggregate of three (3) categories of SAs. The three categories of SAs correspond to the three kinds of communications, best seen from the point of view of the Receiver (Member). Figure 3 depicts this concept:

- *Category-1 SA:*

a SA is required for (bi-directional) unicast communications between the GCKS and a group member (be it a Sender or Receiver). This SA is established only between the GCKS and a Member. In the SMuG Reference Framework, the GCKS entity is charged with access control to the group keys, with policy distribution to members (or prospective members), and with group key dissemination to Sender and Receiver members. This use of a (unicast) SA as a starting point for key management is common in a number of group key management environments [HH99a, HH99b, CP99, RFC2627, BMS99, Bris99].

Note that this (unicast) SA is used to protect the other elements of the GSA (such as the other following two categories of SAs), either in a push or pull model. As such, this SA is crucial and is inseparable from the other two SAs as the definition of a GSA.

From the perspective of one given GCKS, there are as many unique Category-1 SAs as there are members (Senders and/or Receivers) in the group. Thus there may be a scalability concern for some applications, so a Category-1 SA may be used on-demand whereas Category-2 and Category-3 SAs are established at least for the life of the sessions that they support.

- *Category-2 SA:*

a SA is required for the multicast transmission of key-management/control messages (unidirectional) from the GCKS to all group members. As such, this SA is known by the GCKS and by all members of the group.

This SA is not negotiated, since all the group members must share it. Thus, the GCKS must be the authentic source and act as the sole point of contact for the group members to obtain this SA.

From the perspective of each participant in a group (GCKS and all members), there is at least one (1) Category-2 SA for the group.

Note that this allows for the possibility of the GCKS deploying multiple Category-2 SA for other security management purposes.

- *Category-3 SA:*

one or more SAs are required for the multicast transmission of data messages (unidirectional) from the Sender to other group members. This SA is known by the GCKS and by all members of the group.

Similarly, regardless of the number of instances of this third category of SA, this SA is not negotiated. Rather, all group members obtain it from the GCKS. The GCKS itself does not use this category of SA since it is assumed that the GCKS does not transmit data (content) messages.

From the perspective of the Receivers, there is at least one Category-3 SAs for the member-sender (one or more) in the group. This allows for the possibility of including group IDS (GID) in transmission of data packets from the senders in the group.

There are a number of possibilities with respect to the number of Category-3 SAs and the use of GIDs:

- (i) Each sender in the group could be assigned a unique Category-3 SA, thereby resulting in each receiver having to maintain as many Category-3 SA as there are senders in the group.
- (ii) The entire group deploys a single Category-3 SA for all senders, together with the use of GIDs. Receivers would then be able to filter based on the GIDs, whilst maintaining only one Category-3 SA.
- (iii) A combination of(i) and (ii) above.

4.3 Forward and Backward Rekey

The re-key operation is needed to ensure that messages sent to the group cannot be accessed by a former member whose membership has been revoked by the GCKS; some applications may also require that a member who joins a group be denied access to messages that were sent to the group prior to its membership [CP99, HH99a, BMS99]. We call the first case, forward rekey, when a key change is prompted by a member leaving the group. The latter is called backward rekey, when a re-key is caused by a new member joining the group. Note that the terms “forward/backward secrecy” and “forward/backward security” have been used interchangeably in the literature [CP99, HH99a, BMS99, HH99b].

5. SUMMARY AND FUTURE WORK

This paper has described the Group Security Association (GSA) Management model and protocol as developed in the Secure Multicast Group (SMuG) in the IETF.

In order to progress to the point of worthy specifications and working implementations, several questions, among others, must be answered:

- What framework should be used for the group key management building block?
- How many of each category of SA should be allowed in a GSA?
- What transport should be used for Category-2 SA key management control messages?

The first question asks whether the Internet key management framework, ISAKMP, should be used or whether some invented framework should be used to express, specify, and/or implement group key management.

The second question that must be answered is how many SAs of Category-2 and Category-3 must the group key management framework support? This issue has ramifications for how complex the framework will be in terms of messages and payloads. Multiple Category-3 SAs, for example, may be used to bundle keying material for multiple, related groups such as for multimedia sessions [RFC 1889]. A related question concerns GSA updates: are operations needed to modify existing SAs? Such operations may be very complex and may entail changes to group policy, which may have significant ramifications on access control. Re-key algorithms such as LKH and OFT update SAs by modifying keys. Whereas TLS supports operations to change the cipher, IKE requires that a new SA be created and the old SA deleted as the means by which an SA is modified.

The third question is the transport to be used for Category-2 SA messages which are multicast and which have reliability requirements. Should a reliable multicast services be assumed? Should it be integrated into the protocol? More consideration is needed on the effects of providing a multicast key management services to groups of members, large and small, static and dynamic.

These and other questions will be addressed in the near future in the Secure Multicast Group (SMuG) in the IETF.

Acknowledgements

The authors thank various active members of SMuG in the IETF for comments and insights during the development of this work. In particular,

we thank Brian Weis (Cisco) and Cathy Meadows (NRL) for feedback and analysis of the GSA definition and model reported in the current work.

References

- [BF99] B. Briscoe, I. Fairman, Nark: Receiver-based Multicast, Non-repudiation and Key Management, *Proceedings of ACM E-Commerce'99*.
- [BMS99] D. Balenson, D. McGrew, A. Sherman, *Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization*, <http://www.ietf.org/internet-drafts/draft-balenson-groupkeymgmt-oft-00.txt>, February 1999, Work in Progress.
- [BR93] M. Bellare, P. Rogaway, Entity Authentication and Key Distribution, *Advances in Cryptology - Crypto '93, Proceedings*, Springer-Verlag, 1993.
- [Bris99] B. Briscoe, MARKS: Zero Side Effect Multicast Key Management using Arbitrarily Revealed Key Sequences, *Proceedings of NGC'99*.
- [CP99] R. Canetti and B. Pinkas, *A taxonomy of multicast security issues*, <http://www.ietf.org/internet-drafts/draft-irtf-smug-taxonomy-01.txt>, April 1999, Work in Progress.
- [DVW92] Diffie, P. van Oorschot, M. J. Wiener, Authentication and Authenticated Key Exchanges, *Designs, Codes and Cryptography*, 2, 107-125 (1992), Kluwer Academic Publishers.
- [FS00] N. Ferguson and B. Schneier, *A Cryptographic Evaluation of IPsec*, CounterPane, <http://www.counterpane.com/ipsec.html>.
- [HCB00] T. Hardjono, R. Canetti, M. Baugher, P. Dismore, *Secure IP Multicast: Problem areas, Framework, and Building Blocks*, <http://www.ietf.org/internet-drafts/draft-irtf-smug-framework-01.txt>, Work in Progress, 2000.
- [HCD99] T. Hardjono, B. Cain, N. Doraswamy, *A framework for group key management for multicast security*, <http://www.ietf.org/internet-drafts/draft-ietf-ipsec-gkmframework-01.txt>, July 1999, Work in Progress.
- [HH99a] H. Harney, E. Harder, *Multicast Security Management Protocol (MSMP) Requirements and Policy*, draft-harney-msmp-sec-00.txt, March 1999, Work in Progress.
- [HH99b] H. Harney, E. Harder, *Group Secure Association Key Management Protocol*, <http://search.ietf.org/internet-drafts/draft-harney-sparta-sakmp-sec-00.txt>, April 1999, Work in Progress.
- [Kraw96] H. Krawczyk, SKEME: A Versatile Secure Key Exchange Mechanism for Internet, *Proceedings of Network and Distributed Systems Security (NDSS)*, San Diego, 1996.

- [RFC1889] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, January 1996.
- [RFC2093] Harney, H., and Muckenhirn, C., *Group Key Management Protocol (GKMP) Specification*, July 1997.
- [RFC2094] Harney, H., and Muckenhirn, C., *Group Key Management Protocol (GKMP) Architecture*, July 1997.
- [RFC2401] S. Kent, R. Atkinson, *Security Architecture for the Internet Protocol*, November 1998
- [RFC2407] D. Piper, *The Internet IP Domain of Interpretation for ISAKMP*, November 1998.
- [RFC2408] D. Maughan, M. Shertler, M. Schneider, J. Turner, *Internet Security Association and Key Management Protocol*, November 1998.
- [RFC2409] D. Harkins, D. Carrel, *The Internet Key Exchange (IKE)*, November, 1998.
- [RFC2412] H. Orman, *The OAKLEY Key Determination Protocol*, November 1998.
- [RFC2522] P. Karn, W. Simpson, *Photuris: Session-Key Management Protocol*, March 1999.
- [RFC2627] D. M. Wallner, E. Harder, R. C. Agee, *Key Management for Multicast: Issues and Architectures*, September 1998.
- [SDNS88] H. L. Rogers, An Overview of the CANEWARE Program, *10th National Security Conference*, National Security Agency, 1988.
- [SMuG00] www.securemulticast.org
- [WL98] C.K.Wong, S.S. Lam, Digital Signatures for Flows and Multicasts, *Proceedings of IEEE ICNP'98*, October, 1998.

COMMUNICATION-EFFICIENT GROUP KEY AGREEMENT

Yongdae Kim

Dept. of Information and Computer Science

University of California Irvine

kyongdae@ics.uci.edu

Adrian Perrig

Computer Science Division

University of California Berkeley

perrig@cs.berkeley.edu

Gene Tsudik

Dept. of Information and Computer Science

University of California Irvine

gts@ics.uci.edu

Abstract

Traditionally, research in secure group key agreement focuses on minimizing the computational overhead for cryptographic operations, and minimizing the communication overhead and the number of protocol rounds is of secondary concern.

The dramatic increase in computation power that we witnessed during the past years exposed network delay in WANs as the primary culprit for a negative performance impact on key agreement protocols.

The majority of previously proposed protocols optimize the cryptographic overhead of the protocol. However, high WAN delay negatively impacts their efficiency.

The goal of this work is to construct a new protocol that trades off computation with communication efficiency. We resurrect a key agreement protocol previously proposed by Steer et al. We extend it to handle dynamic groups and network failures such as network partitions and merges. The resulting protocol suite is provably secure against passive adversaries and provides key independence, i.e. a passive adversary who knows any proper subset of group keys cannot discover any other group key not included in the subset. Furthermore, the protocol is simple, fault-tolerant, and well-suited for high-delay wide area network.

Keywords: Peer group key agreement, fault-tolerant protocol

1. INTRODUCTION

The proliferation of applications, protocols and services that rely on group communication prompts the need for group-oriented security mechanisms (in addition to the traditional requirements of fault-tolerance, scalability, and reliability). Current group-oriented applications include IP telephony, video conferencing, collaborative workspaces, interactive chats and multi-user games. The security requirements of these applications are fairly typical, e.g., confidentiality, data integrity, authentication and access control. These are achieved through some form of group key management.

The peer nature of many group applications results in certain unique properties and requirements. First, every member in a peer group is both a sender and a receiver. Second, peer groups tend to be small, with fewer than a hundred members. Also, peer groups have no hierarchy and all members enjoy the same status. Therefore, solutions that assign greater importance to some group members are undesirable, since privileged members might behave maliciously; they are also attractive targets of attacks. This essentially rules out the traditional key distribution paradigm as it calls for higher trust in the group member who generates and distributes keys. Finally, since all networks are prone to faults and congestion, any subset of group members must be prepared to function as a group in its own right. In other words, if a network partition splits the members into multiple subgroups, each subgroup must quickly recover and continue to function independently.

In the last two decades a lot of research has been conducted with the aim of minimizing cryptographic overhead in security protocols. It has been long held as an incontrovertible fact that heavy-weight computation – such as large number arithmetic which is the basis of many modern cryptographic algorithms – is the greatest burden imposed by security protocols. We believe that, although this has been the case in the past, rapid advances in computing have resulted in drastic improvements in large-number arithmetic computations. For example, three years ago, a top-of-the-line RISC workstation performed a 512-bit modular exponentiation in around 24 ms. Today, an 850 Mhz Pentium III PC (priced at 1/5-th of the old RISC workstation) performs the same operation in under 1 ms.

In contrast, communication latency has not improved appreciably. Network devices and communication lines have become significantly faster and cheaper. However, the communication (especially via the Internet) has become both accessible and affordable which resulted in drastic increase in the demand for network bandwidth. Consequently, the explosion in the number of users and their devices often causes network congestion and outages. Moreover, while computation power and bandwidth are increasing, network delay is still faced with a fundamental limit dictated by the speed of light.

The bottleneck shift from computation to communication latency leads us to start looking at cryptographic protocols in a different light: allowing more liberal use of cryptographic operations while attempting to reduce the communication overhead. The latter includes both round and message complexity. Communication overhead is especially relevant in a peer group setting since group members can be spread throughout a large network, e.g., the global Internet.

We consider a protocol suggested by Steer et al. in 1988 [SSDW88], one of the first group key agreement protocols. Their protocol is based on the Diffie-Hellman key exchange and assumes the formation of a secure **static group**. We extend their protocol to deal with dynamic groups and network failures. This protocol – referred to as STR hereafter – was neglected due to its heavy computation and communication requirements: $O(n)$ communication rounds and $O(n)$ cryptographic operations are necessary to establish a shared key in a group of n members. However, we extend STR and construct new communication-efficient protocols that support dynamic groups. More concretely, we construct an entire group key management protocol suite, that is particularly efficient in a WAN environment where moderate to high network delays dominate. An extended version of this paper that provides more detail of our algorithms and security is available from the authors.

2. RELIABLE GROUP COMMUNICATION AND GROUP KEY AGREEMENT

In this section, we set the stage for the rest of the paper with a brief overview of the notable features of reliable group communication and group key agreement.

2.1. RELIABLE GROUP COMMUNICATION SEMANTICS

Many modern collaborative and distributed applications require a reliable group communication platform. Current reliable group communication toolkits generally provide one (or both) of two strong group communication semantics: Extended Virtual Synchrony (EVS) [MAMSA94] and View Synchrony (VS) [FLS97]. Both semantics guarantee that: 1) group members see the same set of messages between two sequential group membership events, and, 2) the sender's requested message order (e.g., FIFO, Causal, or Total) is preserved. VS offers a stricter guarantee than EVS: Messages are delivered to all recipients in the same membership as viewed by the sender application when it originally sent the message. In the context of this paper we require the underlying group communication to provide VS. However, we stress that VS is needed for the sake of fault-tolerance and robustness; the security of our protocols is in no way

affected by the lack of VS. More details on the interaction of key agreement protocols and reliable group communication are addressed in [AAH⁺00].

2.2. COMMUNICATION DELAY

Due to the reliable group communication platform, network delay is amplified by the necessary acknowledgments between the group members. The speed of light puts a lower bound on the minimum network delay. For example, a laser pulse that travels through a fiber takes ≈ 10 ms between New York and San Francisco, ≈ 21 ms between Paris and San Francisco, and ≈ 40 ms from London to Sydney. In practice the networks today are slower than the lower bound by about a factor of 4 (due to switching overhead, etc.).

To put this into perspective, an 850MHz Pentium III PC performs a single 512-bit modular exponentiation (one of the most expensive, but most basic public key primitives) in under 1 ms. Moreover, the speed of computers continue to increase. Comparing this with the WAN network delay, it is clear that reducing the number of communication rounds is much more important in the long run for an efficient group key agreement scheme than reducing the computation overhead.

2.3. GROUP KEY AGREEMENT

A comprehensive group key agreement solution must handle adjustments to group secrets subsequent to all membership change operations in the underlying group communication system. The following membership changes are considered:

Join occurs when a prospective member wants to join a group.

Leave occurs when a member wants to leave (or is forced to leave) a group. There might be different reasons for member deletion such as voluntary leave, involuntary disconnect or forced expulsion.

Partition occurs when a group is split into smaller groups. A group partition can take place for several reasons, two of which are fairly common:

- Network failure – this occurs when a network event causes disconnectivity within the group. Consequently, a group is split into fragments.
- Explicit partition – this occurs when the application decides to split the group into multiple components or simply exclude multiple members at once.

Merge occurs when two or more groups merge to form a single group:

- Network fault heal – this occurs when a network event causes previously disconnected network partitions to reconnect.
- Explicit merge – this occurs when the application decides to merge multiple pre-existing groups into a single group.

At first glance, events such as network partitions and fault heals might appear infrequent and dealing with them might seem to be a purely academic exercise.

In practice, however, such events are common owing to network misconfigurations and router failures. In addition, in mobile ad hoc (and other wireless) networks, partitions are both common and expected. Moser et al. present compelling arguments in support of these claims [MAMSA94]. Hence, dealing with group partitions and merges is a crucial component of group key agreement.

3. CRYPTOGRAPHIC PROPERTIES

In this section we summarize the desired properties for a secure group key agreement protocol. Following the model of [KPT00], we define six such properties:

- *Weak Backward Secrecy* guarantees that previously used group keys must not be discovered by new group members.
- *Weak Forward Secrecy* guarantees that new keys must remain out of reach of former group members.
- *Group Key Secrecy* guarantees that it is computationally infeasible for a passive adversary to discover any group key.
- *Forward Secrecy (Not to be confused with Perfect Forward Secrecy or PFS)* guarantees that a passive adversary who knows a contiguous subset of old group keys cannot discover subsequent group keys.
- *Backward Secrecy* guarantees that a passive adversary who knows a contiguous subset of group keys cannot discover preceding group keys.
- *Key Independence* guarantees that a passive adversary who knows any proper subset of group keys cannot discover any other group key.

The relationship among the properties is intuitive. The first two (often typically called Forward and Backward Secrecy in the literature) are different from the others in the sense that the adversary is assumed to be a current or a former group member. The other properties additionally include the cases of inadvertently leaked or otherwise compromised group keys. Forward and Backward Secrecy is a stronger condition than Weak Forward and Backward Secrecy. Either of Backward or Forward Secrecy subsumes Group Key Secrecy and Key Independence subsumes the rest. Finally, the combination of Backward and Forward Secrecy yields Key Independence.

In this paper we do not assume key authentication as part of the group key management protocols. All communication channels are public but authentic. The latter means that all messages are digitally signed by the sender using some sufficiently strong public key signature method such as DSA or RSA. All receivers are required to verify signatures on all received messages. Since no other long-term secrets or keys are used, we are not concerned with Perfect Forward Secrecy (PFS) as it is achieved trivially.

4. PROTOCOLS

We now describe the protocols that make up the STR key management suite: join, leave, merge, and partition. All protocols share a common framework with the following features:

- Each group member contributes an equal share to the group key; this share is kept secret by each group member.
- The group key is computed as a function of all current group members' shares.
- As the group grows, new members' shares are factored into the group key while remaining members' shares stay unchanged.
- As the group shrinks, departing members' shares are removed from the new group key and at least one remaining member changes its share.
- All protocol messages are signed by the sender, i.e., we assume an authenticated broadcast channel.

Before describing the protocols in detail, we review the basic STR key agreement protocol and the notation used in the rest of the paper.

4.1. NOTATION

We use the following notation:

n, N	number of protocol parties (group members)
i, j	group member indices: $i, j \in \{1, \dots, N\}$
M_i	i -th group member; $i \in \{1, \dots, N\}$
r_i	M_i 's session random (secret key of leaf node M_i)
br_i	M_i 's blinded session random, i.e. $\alpha^{r_i} \bmod p$
k_j	secret key shared among $M_1 \dots M_j$
bk_j	blinded k_j , i.e. $\alpha^{k_j} \bmod p$
p	large prime number
α	exponentiation base
Tree-specific notation	
$N_{\langle j \rangle}$	Tree node j
$IN_{\langle l \rangle}$	Internal tree node at level l
$LN_{\langle i \rangle}$	Leaf node associated with member M_i
$T_{\langle i \rangle}$	Tree of member M_i
$BT_{\langle i \rangle}$	Tree of member M_i including all of its blinded keys

Figure 1 shows an example of an STR key tree. The tree has two types of nodes: leaf and internal. Each leaf node is associated with a specific group member. An internal node $IN_{\langle i \rangle}$ always has two children: another (lower) internal node $IN_{\langle i-1 \rangle}$ and a leaf node $LN_{\langle i+1 \rangle}$. The exception is $IN_{\langle 1 \rangle}$ which is also a leaf node corresponding to M_1 . (Note that, consequently, $r_1 = k_1$.)

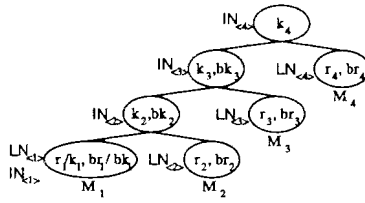


Figure 1 Notation for STR

Each leaf node $LN_{(i)}$ has a *session random* r_i chosen and kept secret by M_i . The blinded version thereof is $br_i = \alpha^{r_i} \bmod p$.

Every internal node $IN_{(j)}$ has an associated secret key k_j and a public blinded key $bk_j = \alpha^{k_j} \bmod p$. The secret key k_i ($i > 1$) is the result of a Diffie-Hellman key agreement between the node's two children. (k_1 is an exception and is equivalent to r_1 .) k_i ($i > 1$) is computed recursively as follows:

$$k_i = (bk_{i-1})^{r_i} \bmod p = (br_i)^{k_{i-1}} \bmod p = \alpha^{r_i k_{i-1}} \bmod p \text{ if } i > 1.$$

The group key in Figure 1 is the key associated with the root node:

$$k_4 = \alpha^{r_4 \alpha^{r_3} \alpha^{r_2 r_1}}$$

We note that the root (group) key is never used directly for the purposes of encryption, authentication or integrity. Instead, such sub-keys are derived from the root key, e.g., by applying a cryptographically secure hash function to the root key. All blinded keys bk_i are assumed to be public.

The basic key agreement protocol is as follows. We assume that all members know the structure of the key tree and their initial position within the tree. (It is simple to have an ordering that uniquely determines the location of each member in a key tree.) Furthermore, each member knows its session random and the blinded session randoms of all other members. The two members M_1 and M_2 can first compute the group key corresponding to $IN_{(2)}$. M_1 computes:

$$\begin{aligned} k_2 &= (br_2)^{r_1} \bmod p = \alpha^{r_1 r_2} \bmod p, & bk_2 &= \alpha^{k_2} \bmod p \\ k_3 &= (br_3)^{k_2} \bmod p, & bk_3 &= \alpha^{k_3} \bmod p \\ \dots & & & \\ k_N &= (br_N)^{k_{N-1}} \bmod p \end{aligned}$$

Next, M_1 broadcasts all blinded keys bk_i with $1 \leq i \leq N - 1$. Armed with this message, every member then computes k_N as follows. (As mentioned above, members M_1 and M_2 derive the group key without additional broadcasts.) Any M_i (with $i > 2$) knows its session random r_i and bk_{i-1} from the broadcast message. Hence, it can derive $k_i = bk_{i-1}^{r_i} \bmod p$. It can then compute all remaining keys recursively up to the group key from the public blinded session randoms: $k_i = br_{i+1}^{k_i} \bmod p$ ($i \leq N$).

Following every membership change, all members independently update the key tree. Since we assume that the underlying group communication system provides *view synchrony* (see Section 2.1), all members who correctly execute the protocol recompute an identical key tree after any membership event. The following fact describes the minimal requirement for a group member to compute the group key:

Remark 1. *If all members know all blinded session randoms of all other members, there exist at least two members who can compute the group key.*

Proof. This follows directly from the recursive definition of the group key. In other words, both M_1 and M_2 (the member at the lowest leaf nodes) can obtain the group key by computing pairwise keys recursively and using blinded session randoms of other members. \square

Remark 2. *Any member can compute the group key, if it knows: 1) its own secret share, 2) the blinded key of its sibling subtree, and, 3) blinded session randoms of members higher in the tree.*

Proof. This also follows from the definition of the group key. To compute the group key, member M_i needs 1) r_i , 2) bk_{i-1} , and 3) $br_{i+1}, br_{i+2}, \dots, br_N$. \square

The protocols described below benefit from a special role (called sponsor) assigned to a certain group member following each membership change. A sponsor reduces communication overhead by performing "housekeeping" tasks that vary depending on the type of membership change. The criteria for selecting a sponsor varies as described below.

4.2. MEMBER JOIN PROTOCOL

We assume the group has n users ($\{M_1, \dots, M_n\}$), when the group communication system announces the arrival of a new member. Both the new member and the prior group receive this notification simultaneously. The new member M_{n+1} broadcasts a join request message that contains its own blinded key bk_{n+1} (which is the same as its blinded session random br_{n+1}). At the same time, the current group's sponsor (M_n) computes a blinded version of the current group key (bk_n) and sends the current tree $BT_{\langle n \rangle}$ to M_{n+1} with all blinded keys and blinded session randoms.

Next, each M_i first increments $n = n + 1$ and creates a new root key node $IN_{\langle n \rangle}$ with two children: the root node $IN_{\langle n-1 \rangle}$ of the prior tree $T_{\langle i \rangle}$ on the left and the new leaf node $LN_{\langle n \rangle}$ corresponding to the new member on the right. Note that every member can compute the group key (see Remark 2):

- All existing members only need the new member's blinded session random
- The new member needs the blinded group key of the prior group
- In a join operation, the sponsor is always the topmost leaf node, i.e., the most recent member in the current group.

As described, the join protocol takes one communication round and two cryptographic operations to compute the new group key (one before the message exchange and one after.)

The join protocol provides backward secrecy since a new member is only given a blinded key of the existing group. However, the protocol does not provide key independence since knowledge of a group key used before the join can be used to compute the group key used after the join. To remedy the situation, we can modify the protocol to require the sponsor to change its session random and the corresponding blinded value, br_n .

4.3. MEMBER LEAVE PROTOCOL

We again have a group of n members when a member M_d ($d \leq n$) leaves the group. If $d > 1$, the sponsor M_s is the leaf node directly below the leaving member, i.e., M_{d-1} . Otherwise, the sponsor is M_2 . Upon hearing about the leave event from the group communication system, each remaining member updates its key tree by deleting the nodes $LN_{\langle d \rangle}$ corresponding to M_d and its parent node $IN_{\langle d \rangle}$. The nodes above the leaving node are also renumbered. The former sibling $IN_{\langle d-1 \rangle}$ of M_d is promoted to replace (former) M_d 's parent. The sponsor M_s selects a new secret session random, computes all keys (and blinded keys) up to the root, and broadcasts $BT_{\langle s \rangle}$ to the group. This information allows all members to recompute the new group key.

In summary, the leave protocol takes one communication round and involves a single broadcast. The cryptographic cost varies depending on two factors: 1) the position of the departed member, and 2) the position of the remaining member who needs to compute the new key.

The total number of serial cryptographic operations in the leave protocol can be expressed as (assuming n is the original group size):

- $2(n - d) + 1 + (n - d) + 1 = 3n - 3d + 2$ when $d > 2$
- $3n - 7$ when $d = 1, 2$

In the worst case, M_1 or M_2 leave the group. The cost for this leave operation is equal to the leave of member M_3 , which is $3n - 7$. The average leave cost is $3n/2 + 2$.

The leave protocol provides forward secrecy since a former member cannot compute the new key owing to the sponsor's changing the session random. The protocol also provides key independence since knowledge of the new key cannot be used to derive the previous keys; this is, again, due to the sponsor refreshing its session random.

4.4. GROUP PARTITION PROTOCOL

A network fault can cause a partition of the group. To the remaining members, this actually appears as a concurrent leave of multiple members. With a minor

modification, the leave protocol can handle multiple leaving members in a single round. The only difference is the sponsor selection. In case of a partition, the sponsor is the leaf node directly below the lowest-numbered leaving member. (If M_1 is the lowest-numbered leaving member, the sponsor is the lowest-numbered surviving member.)

After deleting all leaving nodes, the sponsor M_s refreshes its session random (key share), computes keys and blinded keys going up the tree – as in the plain leave protocol – terminating with the computation of $\alpha^{k_{n-1}} \bmod p$. It then broadcasts the updated key tree $BT_{\langle s \rangle}$ containing only blinded values. Each member including M_s can now compute the group key.

The computation and communication complexity of the partition protocol is identical to that of the leave protocol. The same holds for its security properties.

4.5. GROUP MERGE PROTOCOL

We now describe the STR merge protocol for two groups. (A more general protocol for merging larger number of groups is a straight-forward extension.) We assume that, as in the case of join, the communication system simultaneously notifies all group members (in both groups) about the merge event. Moreover, reliable group communication toolkits typically include a list of all members that are about to merge in the merge notification. More specifically, we require that each member be able to distinguish between the group it was in from the group that it is merging with. This assumption is not unreasonable, e.g. it is satisfied in SPREAD [AAH⁺00].

It is natural to merge the smaller group onto the larger one, i.e., to place a smaller tree directly on top of the larger one. If the two trees are of the same size, we can use an unambiguous ordering to decide which group joins which. (For example, compare the identifiers of the respective sponsors.) Consequently, the lowest-numbered leaf of the smaller tree becomes the right child of a new intermediate node. The left child of the new intermediate node is the root of the larger tree. Since the respective trees are known a priori (before the key management starts), all nodes can construct the new key tree before receiving or computing any cryptographic information.

In the first round of the merge protocol, the two sponsors (topmost members of each group) exchange their respective key trees containing all blinded keys. The highest-numbered member of the larger tree becomes the sponsor of the second round in the merge protocol. Using the blinded session randoms of the other group, this sponsor computes every (key, blinded key) pair upto the intermediate node just below the root node. It then broadcasts the key tree with the blinded keys and blinded session randoms to the other members. All members now have the complete set of blinded keys, which allows them to

compute the new group key. In any case, the merge protocol runs in two communication rounds.

5. ROBUSTNESS

5.1. PROTOCOL UNIFICATION

Although described separately in the preceding sections, the four STR operations: join, leave, merge and partition, actually represent different expression of a single protocol. We justify this claim with an informal argument below.

Obviously, join and leave are special cases of merge and partition, respectively. It is less clear that merge and partition can be collapsed into a single protocol, because in either case, the key tree changes and the remaining group members lack some number (sometimes none) of blinded keys or blinded session randoms which prevents them from computing the new root key. When a partition occurs, the remaining members reconstruct the tree where some blinded keys are missing. In case of a merge, a shorter tree A is merged into a taller tree B . Any member in B now can compute the group key since it knows blinded session random of any member in A . The deepest member in A also can compute the group key since it knows the blinded session random of any other member in A and blinded group key of B . Using the broadcast message any member now can compute the new group key.

We established that both partition and merge initially result in a new key tree with a number of missing blinded keys. In case of merge, the missing blinded keys can be distributed in two rounds. This is because a sponsor in both of A and B broadcasts its own subtree including all blinded keys. Any member in a given subtree can compute the new root key after receiving both broadcasts. The case of partition is very similar except that the missing blinded keys and the new group key can be distributed in one round.

This apparent similarity between partition and merge allows us to lump the protocols stemming from all membership events into a single, unified protocol. The following figure shows the pseudocode.

```
receive msg (msg type = membership event)
construct new tree
while there are missing blinded keys
  if (I can compute any missing keys and I am the sponsor)
    compute missing blinded keys
    broadcast new blinded keys
  endif
  receive msg (msg type = broadcast)
  update current tree
endwhile
```

The incentive for this is threefold. First, unification allows us to simplify the implementation and minimize its size. Second, the overall security and

correctness are easier to demonstrate with a single protocol. Third, we can now claim that (with a slight modification) the STR protocol is self-stabilizing and fault-tolerant as discussed below.

5.2. CASCADED EVENTS

Since network disruptions are random and unpredictable, it is natural to consider the possibility of so-called *cascaded membership events*. In fact, cascaded events and their impact on group protocols are often considered in group communication literature, but, alas, frequently neglected in the security literature. Furthermore, the probability of a cascaded event is much higher on a wide area network. A cascaded event occurs when one membership change occurs while another is being handled. For example, a partition can occur while a prior partition is processed, resulting in a cascade of size two.

We claim that the STR partition protocol is self-stabilizing, i.e., robust against cascaded network events. In general, self-stabilization is a very desirable feature since lack thereof requires extensive and complicated protocol "coating" to either 1) shield the protocol from cascaded events, or 2) harden it sufficiently to make the protocol robust with respect to cascaded events (essentially, by making it re-entrant). The latter is often very complicated and inefficient as seen from [AKNR⁺01].

The pseudocode for the self-stabilizing protocol is shown as below.

```

receive msg (msg type = membership event)
construct new tree
while there are missing blinded keys
  if (I can compute any missing keys and I am the sponsor)
    compute missing blinded keys
    broadcast new blinded keys
  endif
receive msg
if (msg type = broadcast) update current tree
else (msg type = membership event) construct new tree
endwhile

```

Based on view synchrony discussed in Section 2, we provide an informal proof that the above protocol terminates on any finite number of consecutive cascaded events. Due to view synchrony, every member has the same membership view. We can further assume that the ordering of members in the group communication system is same as that of the key tree. By Remark 1, at least a member, say M_i can compute the group key if all of the blinded session randoms are known. All members can then compute the group key using the broadcast message of the member M_i by Remark 2.

Hence, it is enough to show that at least one member knows every other member's session random, eventually. In the above pseudocode, the sponsor is the node below the lowest node whose blinded session random is missing.

Now, if a sponsor M_s cannot compute the group key since some of the blinded keys are missing, it broadcasts the key tree which includes every blinded session random and blinded keys M_s knows. Then the sponsor of the next round will be the one who owns the missing blinded session random. Note that every member will have strictly more blinded session randoms and blinded keys as number of round increases. Hence, as cascaded events stabilize in the group communication system, the STR protocol also terminates.

6. DISCUSSION

6.1. SECURITY

The STR protocol suite and the structure of its group key form a special case of the TGDH key agreement recently presented in [KPT00]. (The latter defines a more general tree-based Diffie-Hellman key agreement.) As such, STR benefits from the provable security of TGDH protocols. Briefly, in [KPT00] it is shown that group key secrecy is reducible to the Decision Diffie-Hellman (DDH) problem [MvOV97].

However, the basic property of group key secrecy is not sufficient for the security of the entire protocol suite. Recall the desired security properties defined in Section 3. We will show that STR offers not only group key secrecy but also weak forward and backward secrecy properties. Furthermore, we show that STR can provide key independence by modifying the protocol slightly.

We now present an informal argument for weak forward and backward secrecy.

The group key secrecy property implies that the group key cannot be derived from the blinded keys alone. At least one secret key K is needed to compute all secret keys from K up to the root key. Hence, we need to show that the joining member M cannot obtain any keys of the previous key tree. First, M picks its secret share r , blinds it and broadcasts α^r as part of its join request. Once M receives all blinded keys on its co-path, it can compute all secret keys on its key path. Clearly, all these keys will contain M 's contribution (r); hence, they are independent of previous secret keys on that path. Therefore, M cannot derive any previous keys.

Similarly, we argue that STR provides weak forward secrecy. When a member M leaves the group, the rightmost member of the subtree rooted at the sibling node changes its secret share. Then, M 's leaf node is deleted and its parent node is replaced with its sibling node. This operation causes M 's contribution to be removed from each key on M 's former key path. Hence, M only knows all blinded keys, and the group key secrecy property prevents M from deriving the new group key.

As presented in Section 4, the STR protocols do not provide key independence. This means that an active attacker who somehow acquires a group key

used before an additive event (join or merge) can use the knowledge of that key to compute a newer key used after such an event. The same does not hold for subtractive events (leave and partition) since a sponsor always changes its session random following each such event.

The join and merge protocols can be modified slightly to provide key independence as explained in the join and merge protocol: Upon each join or merge event, a sponsor (both sponsors, in case of a merge) changes its session random and recomputes its blinded key before proceeding with the rest of the protocol.

This simple change results in key independence since each membership change is followed by at least one session random change. (Of course, we assume that individual members are honest and do not leak their session randoms to the adversary. This behavior can be regarded as equivalent to revealing the group key.)

6.2. COMPLEXITY ANALYSIS

This section compares the computation and communication of STR protocol to other recent group key agreement methods, Cliques GDH.2 [STW00], Tree-Based Diffie-Hellman (TGDH) [KPT00], and Burmester/Desmedt (BD) [BD94]. These protocols provide contributory group key agreement based on different extensions of the two-party Diffie-Hellman key exchange. Moreover, they all support dynamic membership operations.

We consider the following costs:

- **Number of rounds:** this affects serial communication delay. Total number of messages: as the number of messages grows, the probability of message loss or corruption is increased, and so is the delay.
- **Number of unicasts and broadcasts:** a broadcast is much more expensive operation than a unicast, since it requires many acknowledgments within the group communication system.
- **Number of serial exponentiation:** this is the main factor in the computation overhead.
- **Robustness:** Lack of robustness requires additional measures to make the secure group communication system robust against cascaded (nested) faults and membership events.

Table 1 shows a comparison of the current approaches for group key management. The bold text refers to a parameter that severely slows down the protocol in a WAN deployment, for which STR is best suited.

In Cliques GDH.2 protocol, the number of new members k is considered, since the merge cost depends on number of new members. The cost for TGDH is the average value when the key tree is fully balanced. The partition or leave cost for STR is computed on average, since it depends on the depth of the

lowest-numbered leaving member node. For security reasons [STW00], BD always has to restart anew upon every membership event.

As seen from the table, STR is minimal in communication on every membership event. We showed in Section 5 that robustness in the STR protocol is not only easier to implement than in other protocols, but it also achieves higher robustness to network partitions. Cliques GDH.2 is quite expensive protocol in wide area network, since: 1) it is hard or very expensive to provide robustness against cascaded events [AKNR⁺01] and 2) communication cost for merge increases linearly as the number of new members does. In TGDH, the partition protocol is expensive (relatively slow) which may cause more cascaded faults and long delays to agree on a key. The cost of BD is mostly acceptable but large number of simultaneous broadcast messages can be problematic over a wide area network.

Table 1 Protocol Comparison

		Rounds	Messages	Ucast	Bcast	Exp.	Robust
Cliques	J	2	2	1	1	2n	Hard
	L/P	1	1	0	1	n	
	M	k + 3	n + 2k + 1	n + 2k - 1	2	n + 2k	
TGDH	J/M	2	3	0	3	2 log n	Easy
	L	1	1	0	1	log n	
	P	O(log n)	O(log n)	0	O(log n)	O(log n)	
BD		2	2n	0	2n	3	Easy
STR	J	1	2	1	1	2	Easy
	L/P	1	1	0	1	$\frac{3n}{2} + 2$	
	M	2	3	2	1	3k	

J: Join, L: Leave, P: Partition, M: Merge, Ucast: Unicast, Bcast: Broadcast, Exp: Exponentiation

References

[AAH⁺00] Y. Amir, G. Ateniese, D. Hasse, Y. Kim, C. Nita-Rotaru, T. Schlossnagle, J. Schultz, J. Stanton, and G. Tsudik. Secure group communication in asynchronous networks with failures: Integration and experiments. In *ICDCS 2000*, April 2000.

[AKNR⁺01] Y. Amir, Y. Kim, C. Nita-Rotaru, J. Schultz, J. Stanton, and G. Tsudik. Exploring robustness in group key agreement. In *ICDCS 2001*, 2001.

[BD94] M. Burmester and Y. Desmedt. A Secure and Efficient Conference Key Distribution System. In *EUROCRYPT94*, 1994.

[FLS97] A. Fekete, N. Lynch, and A. Shvartsman. Specifying and using a partitionable group communication service. In *ACM PODC '97*, Santa Barbara, CA, August 1997.

[KPT00] Y. Kim, A. Perrig, and G. Tsudik. Simple and fault-tolerant key agreement for dynamic collaborative groups. In *ACM CCS 2000*, November 2000.

[MAMSA94] L. Moser, Y. Amir, P. Melliar-Smith, and D. Agarwal. Extended virtual synchrony. In *ICDCS '94*, June 1994.

[MvOV97] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

- [SSDW88] D. Steer, L. Strawczynski, W. Diffie, and M. Wiener. A secure audio teleconference system. In *CRYPTO '88*, 1988.
- [STW00] M. Steiner, G. Tsudik, and M. Waidner. Cliques: A new approach to group key agreement. *IEEE TPDS*, August 2000.

Going Beyond MAC and DAC Using Mobile Policies

Amgad Fayad, Sushil Jajodia, Don Faatz, Vinti Doshi
The MITRE Corporation

Key words: Access control, Attribute Certificates, DBMS Security, Middleware Security,
Mobile Policy, PKI

Abstract:

Many access control requirements cannot be automated using traditional mandatory access control (MAC) and discretionary access control (DAC) security mechanisms. Examples include user-attribute-based access control and owner-retained access control for handling specially marked data. While several researchers have identified the need for access controls that provide more flexibility than MAC and DAC, the proposed mechanisms for implementing these controls have several shortcomings. In this paper, we describe an access control mechanism that combines attribute certificates with mobile policy to overcome these shortcomings. Attribute certificates permit fine-grained authorisations based on user attributes, such as group membership, rank, and role. Mobile policies allow application-specific policies to move along with the object to other elements of the system. Mobile policies are expressed using an extension to a high-level definition language that we previously proposed in Reference [5].

1. INTRODUCTION

Traditionally, mandatory access control (MAC) and discretionary access control (DAC) security mechanisms have been used for providing information protection. MAC groups users and data based on classification levels. Users can only access data that is classified at their level or lower. Sharing of classified data with users who do not have the required classification clearance is strictly prohibited. In comparison, DAC restricts access to information based on the user's identity and authorisations stating the accesses each user can execute on the objects of the system.

Many access control requirements within the United States (U.S.) Department of Defense (DoD) and the Intelligence Community cannot be automated using traditional MAC and DAC mechanisms. In particular, these communities need to enforce, within an automated system, the true intent of access control requirements associated with special markings, including dissemination controls and release markings, caveats, and warnings [1-3,7,13]. Particular examples include *user attribute-based access control* and *owner-retained access control* for handling specially marked data [13].

Release markings such as Not Releasable to Foreign Nationals (NOFORN) and Top Secret Releasable to Canada (TS REL CANADA) are examples of user attribute-based access control. An object with the NOFORN marking can only be released to an U.S. citizen. Similarly, an object marked with TS REL CANADA can only be given to an individual who has a TS clearance from either the U.S. or Canada.

Originator Controlled (ORCON) release represents an example of owner-retained access control. Any object marked ORCON may be released to users belonging to a specified list of organisations or users; any release to others not on the list requires permission of the originator of the object.

While requirements such as REL XX and ORCON can be implemented using MAC, the solution is cumbersome and, therefore, not an acceptable general-purpose solution. The solutions proposed by others [1-3,13] also have several shortcomings. We will elaborate on this topic in Sections 2 and 3.

Some of the difficulties in carrying out the intent of the aforementioned access control requirements are that a number of these markings are based on user attributes, provisions are made for exceptions, information cannot sometimes be released with the consent of the originator, and the composition of labels when data with different markings are combined is not straightforward.

In this paper, we describe an alternative solution that combines attribute certificates (ACs) with mobile policy. Public key certificates, also known as identity certificates (ICs), allow the identities of the users to be mobile. A user can prove his/her identity, for access control purposes, using his/her IC. The advent of ACs extends ICs beyond identity-based authorisations. That is, attribute certificates provide the capability of assigning *attributes* to users, and, therefore, they are ideal for providing user attribute-based dissemination control.

Mobile policies allow access control rules to move with the objects to which the policies apply. Enforcement can take place within any trustworthy component of the system. We extend the high-level definition language called *Mobile Policy Language* (MPL) (pronounced “maple”) proposed by us [5] to specify policies such as NOFORN and ORCON. MPL extends the traditional Structured Query Language (SQL) access control commands to incorporate attribute certificate information as well as *provisions* [10] that specify required actions to be taken during policy enforcement.

The remainder of this paper is organised as follows. Section 2 describes some motivating examples, Section 3 describes solutions from previous work, Section 4 describes ACs and MPL, Section 5 describes the mobile policy framework, Section 6 shows how to apply MPL and attribute certificates to address the examples from Section 2, and, finally, Section 7 provides conclusions and describes future work.

2. MOTIVATING EXAMPLES

In this section, we take several examples from [13] to illustrate the need for flexible access control policies that go beyond traditional MAC and DAC security mechanisms.

Example 1 Release Markings: NOFORN is a marking used by the DoD/Intelligence community to indicate that access to data requires U.S. citizenship. It is another access control in addition to the restrictions imposed by MAC. Exceptions are often made to the NOFORN control, generally with the use of the REL markings. For example, if a document is marked “NOFORN/REL CAN”, only U.S. and Canadian citizens can access that document. One way to handle NOFORN data within a system’s MAC controls is to define a compartment in the system for NOFORN data. This compartment would be added to each user’s clearance credentials to give the user access to NOFORN data. However, this action may lead to undesirable results, as individuals would then become privy to all data marked with NOFORN control rather than just selected data objects. Also, every time a special-case release needs to be accommodated, a cumbersome process involves adding new compartments and going through the process of changing a user’s clearance.

Example 2 Originator Controlled: Another type of access control used in the IC is ORCON. When ORCON is specified, only users or groups specified by the originator of the data are allowed access. If any additional users or organisations require access to the data, prior consent of the originator is required. For example, suppose a user x from department X releases an object O , marked with ORCON, to users in department Y . Any copy of O made by any user y in department Y would be subject to the same restrictions as object O . Object O (or its copy) is not releasable by users in department Y to users in other departments without the permission of user x , the originator.

Example 3 Label Composition: When two data objects with NOFORN-REL or ORCON markings are combined, how does an automated system combine these markings? For example, suppose object O_1 ’s label states that all government employees and non-government employees cited in access control list A (ACL- A) are granted access to O_1 . Also suppose that O_2 ’s label states that all U.S. citizens and foreign nationals listed in ACL- B are granted access to O_2 . We will call the object O_3 , which is the result of combining O_1 and O_2 . The composite policy for O_3 should state that for a user to have access, the user would need to satisfy one of the following requirements:

- A U.S. government employee with U.S. citizenship
- A U.S. government employee who is not a U.S. citizen and has been given access on ACL- B

- A U.S. citizen non-government employee who has been given access on ACL-A
- A foreign national non-government employee who has been given access on ACL-A and ACL-B

Traditional access control mechanisms based on MAC and DAC are not adequate to automate the policies in these examples. We discuss other solutions and their shortcomings in the next section.

3. PREVIOUS SOLUTIONS

In this section, we describe the solutions provided by McCollum et al. [13] and Abrams et al. [1-3]. We also identify the shortcomings of their solutions.

Since NOFORN and ORCON markings rely on the user's *attributes*, not just clearance level, and ORCON requires specifying an ACL, McCollum et al. [13] introduced user attribute-based access control and Owner-Retained Access Control (ORAC).

User attribute-based access control calls for associating attributes to users. The following are examples of user attributes:

- NOFORN is an attribute that describes U.S. citizens
- CONTRACT is an attribute that describes a government employee
- NOCONTRACT is an attribute that describes a non-government employee

To provide attribute-based access control, McCollum et al. [13] make two assumptions: First, attributes must be assigned in data object labels. Second, access control rules must specify those attributes required for access. While this is a good paradigm, it stops short of discussing how attribute-based access control would be implemented in a distributed environment. Also, it does not provide a mechanism for handling caveats and exceptions to general rules. Finally, it provides no solution to handle exporting data objects from one system to another.

The solution in [13] defines ORAC to enforce ORCON. ORAC calls for using ACLs to allow or explicitly deny access to data objects. The user who

creates the data object is considered its owner and has the right to define an ACL on the object. In contrast to DAC ACL, an ORAC ACL, with its associated owner, propagates along with the data. For example, suppose user *x* creates object1 with ACL-*x*. Suppose further that user *y* copies object1 to produce object2 with ACL-*y*. Object2 will retain ACL-*x* in addition to ACL-*y*.

When enforcing ORCON through ORAC, every time a subject wants to give data access to a new subject who is not on the original ACL, the owner of the data must explicitly approve the access. While this procedure is required in some cases, in other instances the owner may wish to grant another user the right to grant access to the object. Reference [13] suggests a solution using a new parameter, *owner privilege*, which like other privileges can be granted to a subject, and by doing so the owner relinquishes his/her ownership of the data object to others. In this paper, we have another option, called *grant option*, which allows an owner to give grant authority to others for some, but not all, of the privileges on the objects.

Abrams et al. [1-3] built a prototype that had the ability to enforce an open set of access control policies. For each policy, the prototype required that a separate policy-specific module be added to the trusted computing base (TCB) that can enforce the policy. This means that before an object is exported to another component, one must ensure that the appropriate module is available at that component.

4. ATTRIBUTE CERTIFICATES AND MOBILE POLICY LANGUAGE

In this section, we begin with an overview of attribute certificates, followed by a description of MPL. We include several examples to show how our framework can incorporate ORCON and REL XX markings.

4.1 Attribute Certificates

In this paper, we assume that user attributes can be made available through ACs [4, 5]. ACs are digital documents that contain a list of

attributes, each of which is an ordered pair (Tag, Value), where Tag is an identifier and Value is a text string.

Examples of attributes are the following:

- (group, NATO)
- (role, Treaty Negotiator)
- (rank, Major)
- (citizenship, Canada)

An AC, as described in [4], is a structure represented in Abstract Syntax Notation 1 (ASN.1) just like an IC. However, an AC does not contain a public key. An AC must be cryptographically linked to an IC and can be used only in conjunction with this corresponding IC. This restriction is necessary, since an AC cannot provide any information about a user's identity.

ACs provide several benefits. First, ACs can be managed, and distributed, using deployed Public Key Infrastructure (PKI) systems. Second, ACs allow user attributes to be mobile within the distributed system.

Directories provide an alternative to ACs for storing and disseminating user attributes in a distributed computing environment. Like ACs, directories can be used in conjunction with ICs. The following list describes a likely scenario for using directories:

- User x authenticates to server S using his/her IC.
- Server S performs a directory lookup using Lightweight Directory Access Protocol (LDAP) to locate user x's attributes.
- Server S performs attribute-based access control to determine access privileges.

4.2 Mobile Policy Language

MPL has three types of statements: **Grant**, **DoNotGrant**, and **MustGrant**. **Grant** gives access to objects to other users. **DoNotGrant** provides the ability to deny stated accesses to others. Finally, **MustGrant** is a *strong* authorisation; it is used to ensure that stated authorisations would be obeyed by the system and to resolve conflicting **Grant** and **DoNotGrant** statements on a particular access [10-12].

Definition 1 (Authorisation Rule): An authorisation rule is a rule of the following form:

Grant <Access Type> **on** <Object>
to <Security Principal >
[with Grant Authority]
[with provision <Provisions>]
where [<Security Principal> **has attribute** <(tag, value), ...> |
 <predicate>]

Authorisation rules allow grantors to give accesses to other security principals. An access may be granted **with Grant Authority**, which permits the grantee to further grant acquired rights to other users. *Provisions* [10] specify required actions to be taken during the policy enforcement. Stated access to an object is allowed after the conditions in the **where** clause have been validated and the provisions have been applied. The term <predicate> in the where clause is any condition that must be satisfied during the evaluation of the access request.

Example 4: Consider the following authorisation rule:

Grant print **on** Balance_Sheet
 to user
with Grant Authority
with provision Add notice “For Accounting Group Only”
where user **has attribute** (group, accounting group) **and**
 (rank, manager of accounting group)

This rule states that a user can print the file Balance_Sheet if he/she is a member of the accounting group and has the rank of an accounting group manager. The notice “For Accounting Group Only” will be added to the printed copy. Since the print permission has been given with grant authority, a grantee can grant the print right to other users by issuing the following command:

Grant print **on** Balance_Sheet
 to Sue
with provision Add notice “For Accounting Group Only”
where Sue **has attribute** (group, accounting group)

This statement says that Sue can print the `Balance_Sheet` as long as she is a member of the accounting group. Sue cannot grant the print privilege to others, however.

Definition 2 (Negative Authorisation Rule): A negative authorisation rule is a rule of the following form:

```
DoNotGrant <Access Type> on <Object>
to <Subject>
[with provision <Provisions> ]
where [<Security Principal> has attribute <(tag, value), ...>
| <predicate>]
```

Negative authorisation rules explicitly deny access to an object by certain security principals. Some applications require explicit negative authorisations.

Definition 3 (Strong Authorisation Rule): A strong authorisation rule is a rule of the following form:

```
MustGrant <Access Type> on <Object>
to <Subject>
[with provision <Provisions> ]
where [<Security Principal> has attribute <(tag, value), ...>
| <predicate>]
```

Strong authorisation rules facilitate the resolution of conflicting authorisations by superseding decisions from other authorisation rules. We illustrate this by an example.

Example 7: Consider the following authorisation rules:

```
Grant read on file1
to user1
with provision Add copyright notice
where user1 has attribute (group, accounts payable)
```

```
DoNotGrant read on file1
To user1
With provision Notify sysadmin
where user1 has attribute (rank, junior)
```

Suppose further that Alice, who is both a member of the accounts payable group and has rank junior, requests access to file1. The two authorisation rules above are in conflict and, therefore, the system requires a policy to resolve such conflicts.

One way to resolve such conflicts would be to have a default policy that always gives denials precedence over positive grants whenever such conflicts arise. In this case, Alice does not get access to file1. The strong authorisation is more general because it can be used to give either the positive authorisation or the negative authorisation precedence over the conflicting authorisation. For example, we can insert a strong authorisation as follows:

MustGrant read on file1
to user1
with provision Notify VP
where user1 **has attribute** (member, accounts payable group)
and (rank, junior)

In this case, Alice will get access to file1, but a notification will be sent to the VP.

5. MOBILE POLICY FRAMEWORK

In our framework, we make the following assumptions about the environment in which the mobile policy will execute:

- The data object and its associated policy are inseparable. That is, whenever an object moves from one component (server) to another component within the distributed computing environment, so does the associated policy. In particular, if an object is copied, the associated policy is copied as well. This requirement can be fulfilled since servers that receive copies of data objects and mobile policies are trusted to enforce the policies without altering or separating them from the data objects. The disadvantage of this trust model is that it increases the size of the Trusted Computing Base (TCB), to include all servers that receive the data object and mobile policy pair.

- To convert policy declarations in MPL into executable form, compilation of MPL to executable code is performed. The resulting code is called a *mobile policy module* [4, 5]. We are currently building a prototype that generates Java mobile policy modules. For an initial implementation of mobile policy modules, we chose Java servlets. Input to the servlets include requestor identity, attributes, and request type (i.e., read/write/execute). The output is a binary Grant or NoGrant. In addition, provisions are executed prior to servlet termination.
- Associated policies for data objects contain two components. The first component is the *policy declaration*, and the second component is the *mobile policy module*. The policy declaration specifies the policy using MPL statements. The twofold purpose of attaching the policy declaration to the object is to allow the recipients to understand the policy associated with the object and to permit modifications of the policy by the recipients of the objects. If the Grant statement in the authorisation rule contains the **with Grant Authority** clause, the recipient can modify the policy declaration and recompile it to generate a new mobile policy module to attach to the data object. The owner of the object has the option of not including the policy declaration; this is the indication to the recipient that policy modifications are not allowed. Recipient compliance with this requirement is insured since all recipients are part of the TCB.
- When the object is accessed, the system guarantees that it will always execute, and enforce, the associated policy.
- When an object O_1 with policy P_1 is copied to another object O_2 , the creator of O_2 has the option of specifying new access control requirements (call it P_2) that will be enforced in addition to P_1 . Therefore, the new policy for O_2 will be P_1 AND P_2 .

6. APPLYING THE MPL TO THE EXAMPLES

In this section, we revisit the examples from Section 2 and show how MPL can be used to express the required policies.

Example 1 Release Markings Continued: Using MPL, we can specify the NOFORN/REL CAN marking as follows:

Grant read **on** Data_Object
to user1
where user1 **has attribute** (Citizenship, U.S.)
or (Citizenship, Canada)

Instead of defining labels that are associated with each data object, MPL uses the attributes of the subjects. The policy we defined requires user attributes (Citizenship, U.S.) or (Citizenship, Canada) to grant access.

Example 2 Originator Controlled Continued: Suppose Jeremy, from Department A, creates a new data object, which we will call Object1. Suppose further that Jeremy specifies the following access rules on Object1:

- Read access to everyone in department B
- Read and write access to Mary from Department B
- Jeremy requires that he must be consulted if any user, not in department B, is to be granted access to Object1

The following Grant statements expresses this policy:

MustGrant read **on** Object1
to user1
where user1 **has attribute** (Department, B)

MustGrant read/write **on** Object1
to Mary

DoNotGrant read **on** Object1
to *
with provision Notify Jeremy

If Mary wants to give access to Object1 to a user from department C, say Kelly, she must contact Jeremy and ask Jeremy to give access to Kelly.

If Mary copies Object1 to a new object, say Object2, Mary becomes the owner of Object2. However, since Jeremy did not grant Mary **Grant Authority**, Object2 carries over the access permissions of Object1 (as stated in Section 5). Mary still cannot grant read access to Kelly because Kelly is not included in Object1's policy, as defined by Jeremy. Mary needs to

notify Jeremy to include Kelly in his policy. If Jeremy agrees to allow access to Kelly, he defines a new policy on Object1 as follows:

MustGrant read on Object1
to Kelly
where Kelly has attribute (Department, Department C)

Mary can exercise ORCON on Object 2 and add the requirement that she be notified whenever Object2 is accessed by Kelley, as follows:

Grant read on Object2
to Kelly
with provision “Notify Mary”
where Kelly has attribute (Department, Department C)

Example 3 Label Composition Continued: In this example, two objects, Object1 and Object2, are joined to produce Object3. The respective policies of Object1 and Object 2 need to be combined so that the composite policy for Object3 preserves access rules of the two parent objects.

Now let us review Object1’s policy, which states that all government employees and those non-government employees who are in ACL-A are granted access to Object1. This can be represented as follows:

Grant read on Object1
to *
where * has attribute (Employer_Type, Government)
or UserID \in ACL-A

Object2’s policy states that all U.S. citizens and those foreign nationals who are in ACL-B are granted access to Object2. This can be represented as follows:

Grant read on Object2
to *
where * has attribute (Citizenship, US) **or UserID** \in ACL-B

The composite policy on Object3 can be represented as follows:

Grant read on Object3
to *
where * {has attribute (Employer_Type, Government)

or UserID \in ACL-A}
and { **has attribute** (Citizenship, US) **or** UserID \in ACL-B }

7. CONCLUSIONS AND FUTURE WORK

In this paper, we provided a flexible solution to security policy needs, such as NOFORN and ORCON, using MPL. By using MPL, we see that any kind of policy can be easily expressed. It is more expressive and flexible than previously defined methods. Our language also gives various options to enforce NOFORN/ORCON labels. It reduces the difficulty of constantly changing the labels associated with data objects. This is accomplished by using ACs. The SQL-like syntax of MPL makes it fairly easy to implement and understand. The provision clause and the **with Grant Authority** in MPL keeps owner control from being too restrictive.

One of the issues identified with the owner privilege approach [11] was of storage and performance overhead. With MPL, storage is not an issue because the policy is mobile and travels with the data; that is, it is not centralised at one site.

The advantages MPL offers include providing an easy path to implementation in a distributed environment using mobile code, providing a powerful mechanism for handling caveats and exceptions to general rules using provisions, and providing capabilities to handle exporting data objects from one system to another along with the policies associated with them.

We are currently building a prototype to demonstrate how to implement access control using mobile policies and ACs. Our prototype will include an AC parser in order to generate attribute information in text form from ACs in ASN.1 form, it will also include a policy module generator that will produce Java mobile policy modules from MPL declarations, finally we will integrate these modules into a web based demonstration.

8. REFERENCES

1. M. D. Abrams,
“Renewed understanding of access control policies”
Proc. National Computer Security Conf., 1993.
2. M. D. Abrams, K. E. Eggers, L. J. LaPadula, and I. M. Olson,
“A generalized framework for access control”
Proc. National Computer Security Conf., October 1990.
3. M. D. Abrams, J. Heaney, O. King, L. J. LaPadula, M. Lazear, and I. M. Olson,
“Generalized framework for access control: Towards prototyping the ORCON policy”
Proc. National Computer Security Conf., October 1991.
4. S. Chapin, S. Jajodia, and D. Faatz,
“Distributed Policies for Data Management Making Policies Mobile”
Proc. 14th IFIP 11.3 Working Conference on Database Security,
Schoorl, Netherlands, August 2000.
5. V. Doshi, A. Fayad, S. Jajodia, and R. Maclean,
“Using Attribute Certificates and Mobile Policies in Electronic Commerce Applications”
Proc. 16th Annual Computer Security Applications Conference,
New Orleans, LA, December 2000.
6. S. Farrell and R. Housley,
“An Internet Attribute Certificate Profile for Authorization”
PKIX Working Group, Internet-Draft, March 2000.
7. T. D. Graubart,
“on the need for a third form of access control”
Proc. National Computer Security Conf., October 1989.
8. R. Housley, W. Ford, T. Polk, and D. Solo,
“Internet Public Key Infrastructure - X.509 Certificate and CRL profile”
RFC 2459, January 1999.
9. ITU-T Recommendation X.509 (1197 E): *Information Technology - Open Systems Interconnection - The Directory: Authentication Framework*, June 1997.

10. S. Jajodia, M. Kudo, and V. S. Subrahmanian,
“Provisional authorizations”
Proc. 1st Workshop on Security and Privacy in E-Commerce,
Athens, Greece, November 2000.
11. S. Jajodia, P. Samarati, V. S. Subrahmanian, and E. Bertino,
“A Unified Framework for Enforcing Multiple Access Control Policies”
Proc. ACM SIGMOD Int’l. Conference on Management of Data,
May 1997, pages 474-485.
12. S. Jajodia, P. Samarati, and V. S. Subrahmanian,
“A logical language for expressing authorizations”
Proc. IEEE Symp. on Research in Security and Privacy,
Oakland, Calif., May 1997, pages 31-42.
13. C. J. McCollum, J. R. Messing, and L. Notargiacomo,
“Beyond the Pale of MAC and DAC – Defining new forms of access
control”
Proc. IEEE Symp. on Security and Privacy, 1990, pages 190-200.

AN ACCESS CONTROL MODEL FOR DATA ARCHIVES

P. Bonatti¹ E. Damiani¹ S. De Capitani di Vimercati² P. Samarati¹

(1) *Dip. di Tecnologie dell'Informazione – Università di Milano - 20163 Crema, Italy*

(2) *Dip. di Elettronica per l'Automazione - Università di Brescia - 25123 Brescia, Italy*

Abstract We present an approach to enforce access control at data archives that need to make their data selectively available on the Web. The paper discusses protection requirements and access control policies for regulating access to the stored data. It presents a model for enforcing access control regulations and a related language for expressing these regulations.

1. INTRODUCTION

Today's society places great demand on the dissemination and sharing of information. With the development and wide spread use of the Internet and the World Wide Web, that allow for convenient electronic data storage and distribution, organizations in the private and public sectors are more and more required to make their data available to the outside world. An ever increasing amount of data is today collected by statistical agencies and census bureaus for analysis and subsequent distribution to the general public or to specific organizations (e.g., research institutions, government offices). *Data producers* can release the data produced directly, as in the case of national statistical institutions, or exploit the mediation of archive institutions (*data publishers*) that collect data from various sources for their subsequent distribution.

This data distribution process is clearly selective: data cannot just be released to anybody. Rather, specific data can usually be released only to specific requesters or under specific conditions [2,8]. For instance, there are sensitive data that can be released only to specific individuals and/or for specific purposes (e.g., health data collected from hospitals and which must be made available to health care institutions or related partners for research purposes). There are data which are subject to embargoes and can be released to the general public only

after a specific time; there are data that can be released only for non-commercial purposes; and data which do not bear sensitivity, but whose release is subject to payment. Many and many other examples can be mentioned, but these few can already give an idea of the variety of protection requirements that may need to be enforced. This situation calls for the need of powerful and flexible access control systems able to capture and enforce the different requirements that the data producers (or publishers) may need to enforce on the data access. While flexible and expressive enough, the access control system should remain simple, easy to manage, and efficient. In particular, we have identified the following characteristics that the access control system should provide.

- The model should support access restrictions based on the typical abstractions used by data producers and publishers, which can define categorizations of users, purposes of use, types of operations, and data objects. These categories should be definable by the data publisher, and hierarchical structures [8] should be supported.
- The model on which the system is based should support restrictions based on conditions on metadata describing (meta)properties of the stored data and the users, which can be represented through profiles maintained at the system.
- The language to express access control rules should have a declarative form. The use of a declarative language makes it easier the task of specifying access restrictions and keeping control over them.
- The language should be simple and expressive. It should be simple to make the management task of specifying and maintaining the security specifications easy, as well as keeping syntax checking time reasonable. It should be expressive to make it possible to specify, in a flexible way, different protection requirements that may need to be imposed on different data.
- Last but not least, the language should be easy to use to nonspecialists in the field. We could imagine that often, the people specifying the security policies will be employees unfamiliar with procedural or logic-based languages. Therefore, while providing expressive power and unambiguity of these paradigms, the language should however be based on a high-level formulation of the access control rules, possibly close to natural language formulation.

Although many access control models and systems have been proposed [11], current proposals do not completely satisfy the characteristics above. For instance, while most regulations by data producers/publishers make data release

conditioned on the use that the recipient will do of the data, use-based restrictions are not supported by current access control systems. While more recent logic-based authorization languages (e.g., [8]) could provide the expressive power to capture these requirements (or be enriched for that), the resulting system would be too complex to use and manage.

In this paper, we present an access control model regulating access to a data archive together with a language for the specification of security requirements. The language allows data publishers (in the case where data are being distributed by the producer directly, the publisher is the producer itself) to state to whom, how, and under which conditions specific data can be accessed. While expressive and flexible enough to capture the different protection requirements that may need to be imposed on the data, the system remains simple and easy to use.

2. DATA MANAGEMENT AT THE ARCHIVE

The data archive maintains data collected from the different producers for their subsequent distribution. Besides these actual data, called *datasets*, the archive also maintains a collection of *metadata* representing information associated with datasets. We describe datasets and metadata in more details.

2.1. DATASETS

Datasets are data collected from the producers for distribution. Usually, they represent statistical information organized via tables (tabular data) and can be in the form of *microdata*, reporting information of individual respondents, or *macrodata*, e.g., aggregates combining data of different respondents [6]. For the purpose of this paper, we consider data to have already undergone the statistical disclosure control necessary to sanitize data by removing explicit identities of the data respondents, or the possibility of inferring them [7, 10]. Datasets can be organized in abstractions defining groups of datasets that can be collectively referred together with a given name. Groups can reflect the file system organization in directories and/or orthogonal abstractions defined by grouping datasets with common characteristics. Dataset groups need not be disjoint and can be nested. Datasets with their groups define a partial order that introduces a hierarchy [8]. This hierarchy can be depicted as a directed acyclic graph whose nodes are the datasets and groups thereof and an arc from node n_1 to node n_2 indicates a *direct* (i.e., explicitly defined) membership of n_1 in n_2 . Figure 1 illustrates an example of datasets hierarchy, where, for simplicity, the datasets leaves are omitted. The hierarchy divides datasets into two groups: `Free_Datasets` (reporting public data) and `Restricted_Datasets` (which cannot be made available to the general public). In turn `Restricted_Datasets` are organized in `EU_Datasets` (reporting statistics of countries within the European Com-

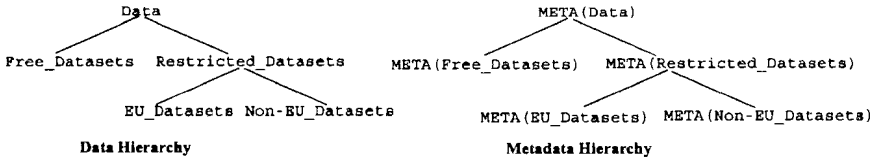


Figure 1 An example of data and metadata hierarchies

munity) and Non-EU_Datasets (reporting statistics of other countries). In the following, we assume the hierarchy to be rooted, meaning there is one element to which all datasets belong. This assumption is not limiting (a dummy node to which all elements belong can be assumed) and is common in many systems [8].

2.2. METADATA

Metadata represent data about data [1]. They are not part of the dataset content; they provide additional contextual information on datasets that can be provided to users and can help them in browsing through the system (searching for specific data). For instance, metadata can report to which study a dataset is referred, how and when it was obtained, by whom, and so on. Although several standards have been proposed for interoperable metadata interchange in the digital libraries domain (e.g., Z39.50, Dublin core, and RDF, [3]) few attempts have been made at employing such standards for the description of statistical information [5]. Rather, information retrieval techniques have been used to extract general-purpose descriptors from available data. In our approach, no assumption is made about metadata syntax and semantics; metadata are assumed to be available in the form of textual or semistructured documents (e.g., XML [1] or DDI [12]). Generic XML-based semistructured documents naturally support heterogeneous metadata formats as they have no fixed structure: the structure can be absent, irregular, or incomplete. Intuitively, a semistructured document can be seen as a set of element properties, possibly nested (element-subelement relationship). Whatever their form, metadata are, at a practical level, files associated with datasets. Metadata are associated only with specific datasets, and not with abstractions on them. Also, no hierarchy is explicitly defined on metadata. However, the abstraction hierarchy defined on the data reflects in an abstraction hierarchy on the corresponding metadata (see Figure 1). We assume a bijective function $META()$ that makes the association between a dataset (or groups thereof) and its metadata (or groups thereof). For instance, given a dataset `dataset1`, function $META(dataset1)$ refers to the metadata associated with it. Given a dataset group `Free_Datasets`, function $META(Free_Datasets)$ denotes the set of all metadata of the datasets in the group. A metadata document can then be referenced either through its identi-

fier or, via function `META`, through the identifier of the dataset with which it is associated.

For metadata browsing by users and (as we will see in Section 4) for the evaluation of conditions that may determine whether or not a given access to datasets can be allowed, it is useful to evaluate the content of metadata. For instance, a user may require access to all datasets produced in the current year (where year is a property in the metadata). The same property can be exploited in the specification of security restrictions by a rule limiting access to datasets produced in the current year to a restricted set of users. For semistructured metadata, we support these features by allowing reference to fine-grained content at the level of properties. Properties (elements and attributes, in the XML terminology) within a metadata document are referenced by means of *path expressions*, stated in an appropriate language, for example XPath [14]. Basically, a path expression is a sequence of element names separated by character / (slash): $l_1/l_2/ \dots /l_n$. Intuitively, semistructured documents can be seen as trees, where each node represents an element or attribute of the considered document and an edge between two nodes represents a containment relationship between them. A path expression $l_1/l_2/ \dots /l_n$ on a document tree then represents all the attributes or elements named l_n that can be reached by descending the document tree along the sequence of nodes named l_1, l_2, \dots, l_{n-1} . For instance, path expression `META(dataset1)/codeBook/stdyInfo/subject` identifies the elements `subject` (describing the topic of a study) within element `stdyInfo` of element `codebook` in the metadata associated with dataset `dataset1`. Path expressions may also include conditions associated with the nodes of a path; in this case the path expression identifies the set of nodes that satisfy all the conditions.

Conditions are distinguished from navigation specifications by enclosing them within square brackets.

For instance, expression `/codeBook//stdyInfo[./subject/keyword = "private schools"] /sumDscr/ [./collDate = "2000-07-05"]` identifies the `sumDscr` element of all the studies whose date of collection is July 5, 2000 *and* one of the salient aspects of the studies's content is `private schools`.

2.3. ACCESSING DATA

Datasets stored at the archive, and metadata associated with them, can be accessed by users via different actions that can be executed on the datasets/metadata. The specific actions supported by a server may vary depending on the functionalities provided on specific kinds of datasets. Among the actions supported, we can distinguish the following three categories (which can correspond to a single action or groups of them):

Browse to visualize and query metadata associated with datasets. With browsing operation, users can walk through the metadata to choose the actual dataset they are interested in.

Analyze-on-line to query datasets. On line analysis includes a set of pre-defined operations that perform on-line calculations on selected data. Available operations may vary depending on the kind of dataset under consideration, and may include basic statistical methods such as: n-way cross tabs, breakdown analysis, correlation, and regression [9].

Download to download data from the server. It allows users to save whole datasets on their local machine to perform off-line analysis.

Further abstractions (or specializations) can be defined on actions, to allow references to groups of actions via a single name. For instance, the three categories of actions above can all be grouped in a set called access and thus referred to as one. In this way, granting a user privilege access to a given dataset will give the user the privilege of executing any action on it.

3. SUBJECT CHARACTERIZATION

We now discuss the characterization of subjects (data requestors) to the purpose of enforcing restrictions on actions that they can execute on the datasets/metadata.

3.1. REQUESTORS

Subjects are entities requesting access to data. The basic concept for the characterization of a subject is the person presenting the request, which is usually referred to as *user*. Users are human entities that can connect to the system and make requests. Each user has associated an identifier (usually the user's login registered at the server), with which the user is referred to in the system.

Although requests are actually typed in by a human user, the decision of whether some data may or may not be released does not depend only on the requesting user's identity but also on the use that the user intends to do of the data being requested, and that can be declared by the user at the time of the request. As a matter of fact, from the analysis of traditional paper world and electronic-based practices at the data archives consulted, it appears clear that the use for which the data are being requested plays an important role in the decision of whether the data can or cannot be released. Although not supported in current access control systems, use-based access restrictions appear to be one of the basic requirements that should be addressed in data dissemination [13]. From an analysis of current practices, we have identified two ways in which the use can be characterized: *purpose* and *project*. Purpose is the reason for which data are being requested and will be used. Examples of

purposes are: Research, Commercial, Teaching, or Personal interest. A project is a named activity registered at the server, for which different users can be subscribed, and which may have one or more purposes. As an example, one or more organizations involved in a given research project can register the project to the archive so that all users working on it (as specified by the authority registering the project at the archive) can, in the execution of the project's activities, enjoy the project's privileges for accessing data maintained at the archive.

Accordingly, we characterize each subject making a request to the data publisher server with a triple $\langle user, project, purpose \rangle$ stating that *user* is requesting an access for a given *project* and/or a given *purpose*. Access requests are then characterized by the subject requesting access, the action requested, and the object on which the action is requested. Some elements within the subject triple may remain unspecified with respect to a given request. This is, for example, the case of requests made by users who do not belong to any project or who do not declare the purpose for which the data are being requested. Identity also can remain unspecified, as in the case of anonymous requests.

Example 3.1 Examples of access requests are as follows.

- $\langle \text{tom.smith, FASTER, research} \rangle, \text{download, dataset1}$
user tom. smith requires to download dataset1 for research purposes within the FASTER project.
- $\langle \text{john.doe, -, commercial} \rangle, \text{download, dataset1}$
user john. doe requires to download dataset1 for commercial purposes.
- $\langle \text{-, -, -} \rangle \text{browse, meta dataset5}$
an anonymous user with undeclared project and purposes requires to browse metadata meta dataset5.

3.2. SUBJECT ABSTRACTIONS AND PROFILING

Besides their identities or declared project and purpose (composing the request), subjects are characterized at the server by additional information, such as membership in groups or satisfaction of given properties, which may affect their ability to access data. We now discuss the definition and organization of subject related information.

3.2.1 Subject abstractions. Abstractions allow the grouping of users, projects, and purposes, respectively, with common characteristics, and referencing to the groups with a name. For instance, with respect to projects, abstractions can group together all the projects registered by a given organization, all the projects sponsored by a national institution, or all the projects

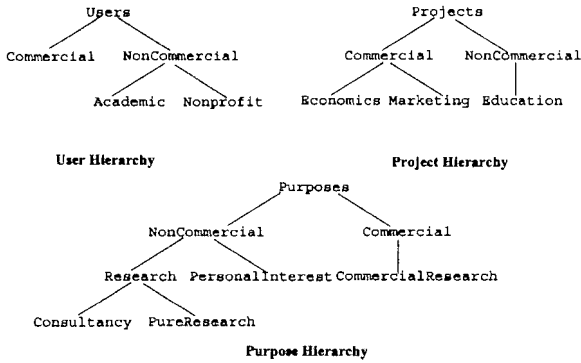


Figure 2 An example of user, project, and purpose hierarchies

with commercial goals. With respect to purposes, abstractions can correspond to generalization/specialization relationships. For instance, pure research and applied research can be seen as a specialization of research. With reference to the user domain, abstractions allow the definition of groups, representing named sets of users, as usually supported in current access control systems [8, 11]. At a very high level, groups can distinguish the different communities of users who may need access to a data archive, such as: academic community, policy making community, mass media community, and commercial community [9]. Specializing these communities, we can obtain finer grained or orthogonal classifications of the users. For instance, going at a finer grain we can distinguish, within the academic community, groups private_schools vs state_schools, or Faculty and Students. At an orthogonal level, we could also classify users based on other aspects such as their citizenship (e.g., EU_citizens vs Non-EU_citizens).

Abstraction groups can be nested (i.e., groups can be defined as members of other groups) and need not be disjoint (e.g., a user can belong to more than one group). The membership relationship between abstraction groups introduces then a hierarchy (partial order) on the domains of users, projects, and purposes. Figure 2 illustrates an example of users, projects, and purpose hierarchies, where, for simplicity, only the abstractions are depicted and leaf nodes (corresponding to individual users, projects, and purposes) are omitted instead.

3.2.2 Users and projects profiles. The data publisher server recognizes only users and projects registered at the server. Each user and project is assigned an identifier that allows the server to refer to the user (project, resp.). Besides their identifiers, users and projects registered at the server usually have other properties associated with them. For instance, a user may have properties such

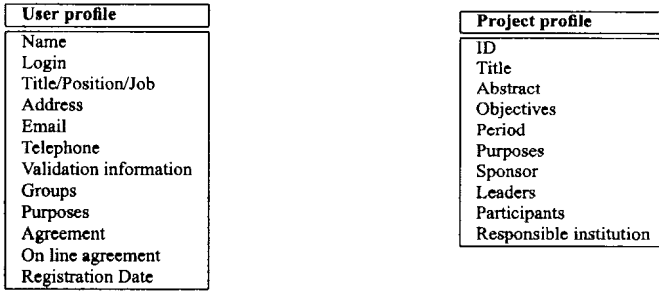


Figure 3 An example of user and project profiles

as name, address, and occupation; a project may have properties such as title, abstract, and sponsor. To capture and reason about these properties we assume each user and project is associated with a *profile*. Intuitively, profiles are to users and projects what metadata are to datasets. Properties in profiles allow the enforcement of access restrictions that traverse group boundaries.¹ To be as general as possible, we view profiles as semistructured documents (XML or RDF like [1]). The profile associated with a user (project, resp.) defines the name and value of the properties that characterize the user (project, resp.). The semistructured format of user and project profiles provides flexibility in the definition of meta properties associated with subjects (e.g., certain properties can be specified only for given classes of users). Figure 3 illustrates an example of profile for users and projects.

4. ACCESS CONTROL RULES AND ACU LANGUAGE

In the previous sections, we have discussed the form of access requests and the organization of data and subject-related information at the server. We now present the rules that establish access regulations for subjects to access data, and a language for expressing them. We start by introducing the components of the rules, we then give their format and semantics.

4.1. SPECIFICATION OF SUBJECT, OBJECTS, AND CONDITIONS

The first step in the specification of access control rules is the characterization of the *subjects*, *actions*, and *objects* to which each rule applies, and of possible *conditions* under which the specific access can be executed. Actions

¹ In principle, every property could be supported through groups, but this would require the definition of as many groups as the cardinality of the property domain, with a result that would be rather awkward and impracticable.

Keywords	CAN WITH IF ONLY IF IN AND OR NOT FORMETA PURPOSES PROJECTS	
Reserved Identifiers	user	bounded to the identity (if defined) of the user making a request
	project	bounded to the project (if defined) specified by the user making a request
	purpose	bounded to the purpose (if defined) specified by the user making a request
	dataset	bounded to the identifier of the dataset to which access is requested
	metadata	bounded to the identifier of the metadata document to which access is requested or associated with the dataset to which access is being requested.

Figure 4 List of keywords and reserved identifiers of the ACU language

are characterized simply through the name of the operation or class of operations (in which case the rule applies to all operations in the class). Subjects and objects can also be specified simply by stating an identifier, specifying a given elementary value in the corresponding domain or a named abstraction of values. To provide expressiveness and flexibility, our language also allows the specification of subjects and objects through expressions, where each expression identifies a set of subjects (objects, respectively) that satisfy specific properties. To make it possible in these expressions to refer to the user, project, purpose, data, or metadata involved in the request being evaluated without need of introducing variables in the language [8], we provide the reserved identifiers listed in Figure 4. The appearance of one of such identifiers (e.g., **user**) in an expression is intended to be replaced with the actual parameter of the request (e.g., user requesting access) in the evaluation at access control time. The value is “undefined” in case no value has been declared. Object and subject expressions can also use the keywords listed in Figure 4. The meaning of some keywords is straightforward (e.g., AND, OR, and NOT are boolean operators, and IN denotes membership in abstraction), the meaning of the others will be clear in the following.

4.1.1 Object expressions. The specification of the objects to which a rule applies is an *object expression* of the form

$$\textit{object-id} \text{ [WITH } \textit{conditional-object-expression} \text{]}$$

where:

- *object-id* is either the identifier of a dataset (or group of datasets) or of a metadata document (or group thereof) together with an optional XPath expression identifying portions of the document. Metadata document can be identified explicitly via their identifier or, via function META, by specifying the name of the datasets (or group thereof) with which they are associated.
- *conditional-object-expression* is a boolean formula of conditions that can evaluate membership of the object in groups, values of properties on metadata, and so on.

Example 4.1 The following are examples of object expressions.

- `Free_Datasets WITH META(dataset)/producer=ACME`
it denotes all datasets in the `Free_Datasets` class that are produced by ACME (the producer is specified as a property in the associated metadata).
- `META(Restricted_Datasets)//question_text`
it denotes element `question_text` within the metadata documents associated with datasets in the `Restricted_Datasets` group.

The use of abstractions, reserved identifiers, and path expressions to query metadata provide a flexible and powerful means of identifying via a simple expression a whole set of datasets/metadata, which will turn very convenient in the specification of access rules [4]. In particular, given the reachness of the metadata usually supported [9], expressions allow the specification of access rules applicable only to datasets whose metadata satisfy some conditions. For instance, it allows the enforcement of embargo restrictions, where only datasets collected before a given year can be released.

4.1.2 Subject expressions. In an analogous way, subjects to which a rule applies are specified as a *subject expression* in the form of a boolean formula of terms that evaluate conditions on the user, project, and purpose of the request. The rule will be applicable only to subjects that satisfy the given conditions, where conditions can evaluate the user's profile or its membership in groups, the project's profile or its membership in a group, and the purpose value or its inclusion in an abstraction. We assume profiles to be referenced with the identity of the corresponding users and projects. Single properties within users and projects profiles are referenced with path expressions denoting the path from the root to the property. For instance, `FASTER/sponsor/address` indicates the address of the sponsor of the `FASTER` project. Here, `FASTER` is the identity of the project (and therefore the identifier for the corresponding profile), and `sponsor/address` the path name of the address property. Expressions can make reference to the user, project, and purpose involved in the current request via the reserved identifiers **user**, **project**, and **purpose**, respectively (see Figure 4).

Example 4.2 Some examples of subject expressions are as follows.

- `user/citizenship=EC AND (project/ sponsor=EC OR purpose IN research)`
it denotes requests made by users who are European citizens and intend to use the data for research purposes or within an EC funded project.
- `user IN NonCommercial - users AND purpose IN research`
it denotes requests made by users belonging to group `NonCommercial -users` that intend to use the data for research purposes.

- **user** IN NonCommercial-users AND **purpose** IN research AND **project** /sponsor=EC
it denotes requests made by users belonging to group NonCommercial-users that intend to use the data for research purposes within an EC funded project.

In the case of security constraints applicable to all users within a given group or that request access for a given project or purpose (or group thereof), the group, project, and/or purpose element can be explicitly factorized out of the subject expression and isolated. The explicit reference to users/groups, projects, and purposes allows the indexing of the ACU rules and consequently improves performances in the access control.

Intuitively, a subject expression of the form

- **user** IN user-id AND **project** IN project-id AND **purpose** IN purpose-id AND subject-expression

can be turned into an indexable expression of the form

- user-id OF project-id PROJECTS FOR purpose-id PURPOSES WITH subject expression

where the clauses “OF project-id PROJECTS”, “FOR purpose-id PURPOSES”, and “WITH subject expression” are optional and can be omitted.

4.1.3 Conditions. Besides subjects, objects, and actions, access control rules can specify *conditions* defining constraints that the rule requires be satisfied for the request to be granted. Conditions evaluate membership of subjects and objects into classes or properties in their profiles and associated metadata. These are conditions similar to those appearing in subject and object conditional expressions, but which may need to be stated separately (as it will be clear in the next subsection).

4.2. ACCESS RULES

Our system supports two kinds of access rules: *authorizations* and *restrictions*.

Authorizations specify permissions for the access. They have the form

$$\langle subjects \rangle \text{ CAN } \langle actions \rangle \langle objects \rangle [\text{ IF } \langle conditions \rangle]$$

where *subjects*, *actions*, and *objects* identify the requests to which the authorization applies as discussed in the previous section, and *conditions* is a boolean expression of conditions whose satisfaction authorizes the access. Note that conditions can also be included in the expressions specifying the *subjects* and

object for the rule. An access request is considered to be authorized if at least one of the authorizations that applies to the request is satisfied.

Restrictions specify requirements that *must* be satisfied for an access to be granted. They have the form

⟨*subjects*⟩ CAN ⟨*actions*⟩ ⟨*objects*⟩ ONLY IF ⟨*conditions*⟩

where *subjects*, *actions*, and *objects* identify the requests to which the restriction applies as discussed in the previous section, and *conditions* is a boolean expression of conditions that every request to which the restriction applies must satisfy; lack to satisfy any of the conditions in restrictions that apply to a given request implies that the request will be denied. Unlike for authorizations, conditions cannot be all incorporated in the subject and object expressions of the rules as this would change the semantics of the restrictions. While conditions appearing in the *conditions* field impose constraints that if not satisfied imply that the access should be denied, conditions in the subject (object) expressions simply limit the requests to which the restriction is applicable. As an example, notice the difference between statements like “Users can access data1 *only if* they are non-commercial and have signed an agreement” and “Users who are non-commercial can access data1 *only if* they have signed an agreement”. While the first rule prohibits access to commercial users, the second rule does not.

Authorizations correspond to traditional (positive) rules usually enforced in access control systems [11]. If multiple authorizations are applicable to a given access request, the request can be granted only if at least the conditions in one authorization are satisfied. Therefore, lack to satisfy the conditions in an authorization applicable to a request simply makes the authorization ineffective; but it does not imply that the access will be denied. Intuitively, this means that different authorizations are considered as combined in OR.

The only support of authorizations (traditional open policy) would result however limiting. As a matter of fact, by looking at the specifications of several partners we noticed that often access restrictions are stated in a *restrictive* form, rather than in the *inclusive* positive form just mentioned. By restrictive form we mean rules that state conditions that *must* be satisfied for an access to be granted and such that, if at least one condition is not satisfied, the access should not be granted. For instance, a rule can state that “access to dataset1 can be allowed *only* to citizens”. It is easy to see that such a restriction cannot be simply represented as an authorization stating that citizens are authorized. In fact, while the single authorization brings the desired behavior, its combination with other authorizations may not, leading the *only* constraint to be not satisfied anymore. The combined use of authorization and restrictions easily support both requirements: restrictions specify requirements of the exclusive *only if* form, while authorizations specify requirements in the traditional positive *if*

form. Intuitively, restrictions play the same role as negative authorizations (denials) supported by recent access control systems (a restriction is equivalent to a negative authorization where the condition is negated). However, we decided to introduce restrictions as their format appears to be closer to the intuitive formulation of protection requirements in the policies examined. Restrictions are also easier to understand because of the clear separation between subjects to which a restriction applies on the one side and necessary conditions that these subjects must satisfy on the other side (which, in traditional approaches, would be collapsed into a single field).

As visible from the example below, the specification of authorizations and restrictions while ensuring non-ambiguity and a clear semantics, results very intuitive and close to the natural language formulation of the requirements.

Example 4.3 The following are examples of security requirements and corresponding ACU rules enforcing them.

Rule 1) Everybody can access `Free_Datasets`.

- `Users CAN access Free_Datasets`

Rule 2) Access to datasets not in `Free_Datasets` allowed *only* to UK citizens.

- `Users CAN access data WITH NOT dataset IN Free_Datasets ONLY IF user/citizenship='UK'`

Rule 3) NonCommercial users can download `Standard_Datasets` if project is Educational and its sponsor is a non-profit organization.

- `NonCommercial-users OF Educational PROJECTS CAN download Standard_Datasets IF project/sponsor='non-profit'`

Rule 4) Users within NonCommercial projects who are employed as faculty members can access `Standard_Datasets`.

- `Users OF NonCommercial PROJECTS CAN download Standard_Datasets IF user/title = 'faculty'`

4.3. ACCESS CONTROL ENFORCEMENT

The Access Control Unit (ACU) component mediates all the access requests to datasets/metadata and evaluates them against the access rules. As already discussed in Section 3, each access request is characterized by three elements: the *subject* that makes the request (composed of the triple $\langle user, project, purpose \rangle$), the *object* on which the request is made, and the *action* that the subject wishes to perform on the object. For each request received, the access control system first determines all the rules that apply to the request, that is, the rules for which the action field is equal or is an abstraction of the action in the request,

and whose subject (object, respectively) expressions are satisfied by the subject (object respectively) of the request. This rule collection process is followed by a conditions packing and evaluation process as follows. All the conditions appearing in the applicable rules are evaluated. According to the given semantics, for the access to be granted *all* the (*only if*) conditions in the restrictions must be satisfied *and* the (*if*) conditions of *at least one* authorization must be satisfied. The system therefore evaluates the satisfaction of the resulting combined condition, substituting true or false for conditions that can be evaluated against profiles and metadata. If the required conditions are satisfied the access is granted, it is denied otherwise.

Example 4.4 Consider the access control rules in Example 4.3 and a request by user Alice to download dataset1 for Commercial purpose within project Al_Marketing. Suppose that dataset1 is a Free_Datasets. The access will be granted with no condition according to rule 1.

Consider now a request by user Bob to analyze on line dataset2 for Research purpose within Educational project. Suppose dataset2 belongs to Standard_Datasets. Authorizations 3 and 4 and restriction 2 apply to the request. Accordingly, the access can be granted only if the conditions in the restriction (**user/citizenship='UK'**) *and* the conditions in at least one of the authorizations (**project/sponsor='non-profit'** or **user/title = 'faculty'**) are satisfied. Suppose that, according to the profile information, Bob is a UK citizen, the sponsor of the project is a non-profit organization (conditions in rules 2 and 3 are true), and Bob is a student (condition in rule 4 is false). Restriction 2 and at least one authorization are satisfied and therefore access is granted.

5. CONCLUSIONS

We have presented a model to regulate access to data to be made available for controlled distribution over the Web. The approach is based on a flexible access control model based on a fully-declarative, simple, and expressive language able to express the different protection requirements that may need to be enforced. We are currently extending the language to the consideration of dynamic conditions (e.g., sign agreements) and support of user-system dialog. A prototype implementation is also being developed.

Acknowledgements

The work presented in this paper has been carried out in the context of the FASTER project (www.faster-data.org), a project funded by the European Community targeted to the design and development of a secure Web-based system for the selective distribution of data on the Web. We would like to thank

all the partners of the FASTER project,² and Titto Assini in particular, for useful discussions on protection requirements of data archives and for providing actual access control policies to be enforced.

References

- [1] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Academic Press/Morgan Kaufmann, 1999.
- [2] P. Bonatti and P. Samarati. Regulating service access and information release on the web. In *Proceedings of 7th ACM Computer and Communication Security*, Athens, Greece, November 2000.
- [3] Communications of the ACM, April 1998.
- [4] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati. Design and implementation of an access control processor for XML documents. *Computer Networks*, 33(1–6):59–75, June 2000.
- [5] C.S. Dippo. Fedstats promotes statistical literacy. *Communications of the ACM*, 41(4):58–60, April 1998.
- [6] N. Kirkendall et. al. Report on statistical disclosure limitation methodology. Statistical policy working paper, no. 22, Washington: Office of Management and Budget, 1994.
- [7] A. Hundepool and L. Willenborg. μ - and τ -Argus: Software for statistical disclosure control. In *Third International Seminar on Statistical Confidentiality*, Bled, MA (USA), 1996.
- [8] S. Jajodia, P. Samarati, M.L. Sapino, and V.S. Subrahmanian. A Unified framework for supporting multiple access control policies *ACM Transactions on Database Systems*, 2000. To appear.
- [9] S. Musgrave and J. Ryssevick. The social science dream machine: Resource discovery, analysis and delivery on the web. In *Social Science Computing Review*. To appear.
- [10] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Data and Knowledge Engineering*, 2001. To appear.
- [11] P. Samarati and S. Jajodia. Data security. In J.G. Webster, editor, *Wiley Encyclopedia of Electrical and Electronics Engineering*. John Wiley & Sons, February 1999.
- [12] The data documentation initiative codebook DTD - version 1.0, March 2000. <http://www.icpsr.umich.edu/DDI/CODEBOOK.TXT>.
- [13] B. Thuraisingham, S. Jajodia, P. Samarati, J. Dobson, and M. Olivier. Privacy issues in www and data mining: Panel discussion. In S. Jajodia, editor, *Database Security XII- Status and Prospects*. Kluwer, 1999.
- [14] World Wide Web Consortium (W3C). *XML Path Language (XPath) Version 1.0*, November 1999. <http://www.w3.org/TR/xpath>.

²Partners of the project are: The Data Archive at University of Essex (UK), University of Milan (Italy), Norwegian Social Science Data Services (Norway), Dansk Data Arkiv (Denmark), Centraal Bureau voor de Statistiek (Netherlands), Central Statistics Office (Ireland), Statistik Sentralbyra (Norway), Centre d'Informatisation des Donnees Socio-Politiques (France).

Checklist-Based Risk Analysis with Evidential Reasoning

Sungbaek Cho and Zbigniew Ciecchanowicz

Information Security Group, Royal Holloway College, University of London, Egham, Surrey, TW20 0EX, UK

Key words: Risk Analysis, Checklist, Belief Function, Evidential Network, BS7799

Abstract: Measuring risk is not a simple task since it almost invariably includes an analyst's subjective judgment. Risk analysis often forces the analyst to estimate or predict future events, which are uncertain. Therefore, we should consider the uncertainties associated with judgments made by the analyst. Hence in this article, we try to apply belief functions, which are used to express and manipulate uncertainties. We use an evidential network to combine answers and uncertainties from a checklist-based risk analysis. A checklist method is still useful in that it is relatively easier and simpler than other risk analysis methods. Furthermore, a checklist-based risk analysis can be used in a baseline approach. To establish the measure of risk in a checklist-based analysis, and the uncertainty that exists in this measurement, we suggest the use of belief functions. An evidential network deployed in a checklist-based risk analysis can also be applied to the self-assessment of BS7799 compliance when preparing for accredited certification against BS7799.

1. INTRODUCTION

Risk analysis is a useful tool for organisations in identifying possible security holes in information systems and providing appropriate countermeasures against them. Risk analysis is, by definition in ISO/IEC TR13335-1 (1996), the process of identifying security risks, determining their magnitude, and identifying areas that need safeguards. Risk analysis is an essential tool for systematic management of information security as it is used in identifying the potential risks and providing useful information to

management for planning and organising security. Pfleeger (1997) argues that there are several benefits of risk analysis such as improved awareness of security, identification of assets, threats and vulnerabilities and improved decision basis for security investment. However, he also points out that the problems in risk analysis, such as imprecise inputs, too much focus on numeric values, and users' tendency to use the same inputs over several years, are the factors that bring into doubt the value of risk analysis. These problems are not just restricted to information security risk assessment, but are equally valid in any complex and unstructured decision making situation. Within risk management, risk analysis is regarded as the point where most difficulty arises (Rainer, et al. 1991).

Besides the methodological difficulties in measuring risk, the other concern is that formal risk analysis can be a time-consuming and expensive process (Erwin1994). Formal risk analysis includes the identification and valuation of assets, threats and vulnerabilities, as detailed in ISO/IEC TR 13335-3 (1997). However, it is equally true that risk analysis is critical for preserving security and the benefits of a well-performed risk analysis far outweigh any drawbacks (Ciechanowicz 1997). From the viewpoint of level of detail and granularity of risk analysis, methods are mainly classified into four categories (ISO/IEC TR 13335-2 1997): (1) baseline approach, (2) informal approach, (3) detailed risk analysis and (4) combined approach. In the baseline approach, a standard set of safeguards is applied to all information systems so as to achieve a baseline level of protection. In an informal approach, we conduct a pragmatic risk analysis on all systems by exploiting the knowledge and experience of security professionals. Detailed risk analysis refers to the detailed review of systems, which includes the identification and valuation of assets, and assessment of the levels of threats to those assets and associated vulnerabilities. The combined approach balances the baseline and detailed approaches by applying detailed risk analysis to important systems while protecting less important systems with a baseline approach. ISO/IEC TR 13335-3 (1997) recommends the use of the combined approach for efficient and effective allocation of organisational resources for risk analysis.

2. CHECKLIST METHOD AND BASELINE APPROACH

Owing to the critical role of risk analysis in security management, a number of risk analysis methods have been developed since the early 1980s. Examples include CRAMM (CCTA Risk Analysis and Management Method, CCTA 1990), annualised loss expectancy (ALE), Courtney, the

Livermore Risk Analysis Method (LRAM), Stochastic Dominance, Checklist, and Fuzzy metrics. An overview of these methods is well summarised in Rainer, et al. (1991).

In this article, our interest is in checklist-based risk analysis. Checklist method (also known as a simple questionnaire method) uses a series of questions to assess risk. There are a number of sources that provide security checklists such as manuals from computer system vendors and publications from security organisations. Examples are BS7799 part 1 & 2 (1999), ISO/IEC TR 13335 Part 4 (1999), IT-Baseline Protection Manual (GISA 1997), and the NIST Handbook (1995). In these source materials, questions and checklists are generally listed by either functional areas such as input, processing and output, or asset types such as hardware, software and personnel. Therefore, we need to convert these generic checklists to specific questions tailored for risk analysis. The advantage of the checklist method is its simplicity in identifying major weaknesses.

The baseline approach is a simple way of performing risk analysis as it consists of (1) listing assets, (2) listing threats associated with each asset, (3) listing vulnerabilities associated with a pair of [asset, threat], (4) identifying existing controls for a triplet of [asset, threat, vulnerability], and (5) collecting all the information and assessing the measure of risk in a simple and pragmatic way (BS7799: Guide to Risk Assessment 1998). Once we have established a set of checklists that assess the vulnerability associated with each pair of asset and threat, the security review based on this set of checklists can be used as a baseline approach. A typical question in such a checklist may be 'given threat j on asset i , is there countermeasure k against threat j ?' The main difference between our suggested approach and the BS7799 baseline approach is that we consider vulnerabilities and countermeasures simultaneously by using predefined checklist questions while BS7799 separates the identification of vulnerabilities and existing countermeasures. The suggested approach provides a much simpler evaluation by considering both vulnerabilities and corresponding countermeasures simultaneously. Although the checklist method does not provide the detailed insight found in a detailed risk analysis, it is still a useful method in that it gives us an overview of the system's security in a reasonably short time period. Also, it is the only applicable method where there is no risk analysis expertise or organisational resource such as budget and time to perform a detailed risk analysis.

One concern in the checklist method is how to manipulate the gathered answers so as to highlight areas that need management attention. Without a highlighting capability, the output of checklist-based risk analysis will be a lengthy list of answers to questions; such a list is of very limited use to management and prevents quick decisions for improving security. The most

common method for solving this problem is the use of a scoring method. In a simple scoring method, one may consider the following scheme:

Let N_i be the number of threats associated with asset i and N_{ij} be the number of vulnerability checkpoints (or questions) for threat j ($1 \leq j \leq N_i$), which is associated with asset i . Assign the vulnerability score S_{ijk} ($1 \leq k \leq \max\{N_{ij} | 1 \leq j \leq N_i\}$) to each applicable vulnerability checkpoint. Assign $S_{ijk} = 0$ where the checkpoint does not exist for a triplet (i, j, k) . Then the measure of risk for asset i , denoted by R_i , can be calculated as follows:

$$R_i = \sum_j \sum_k N_i^{-1} N_{ij}^{-1} S_{ijk}$$

This measure represents the normalised sum of total vulnerability score associated with asset i . The normalisation is required as the number of vulnerability checkpoints and the number of threats varies with the assets and threats, respectively (each threat may have a different number of vulnerability checkpoints and each asset may have a different number of threats). The more advanced scoring method (i.e., the weighted average method) may appear in various forms. Examples include the following two equations.

If the weights (V_{ijk}) that are specific to each threat j are assigned to each vulnerability checkpoint, then R_i in the above example becomes:

$$R_i = N_i^{-1} \sum_j \sum_k V_{ijk} S_{ijk} \text{ where } \sum_k V_{ijk} = 1$$

If the weights (V_{ijk}) that are specific to each threat j are assigned to each vulnerability checkpoint, and the weights (T_{ij}) that are specific to each asset i are assigned to each threat j , then R_i in the above example becomes:

$$R_i = \sum_j \sum_k T_{ij} V_{ijk} S_{ijk} \text{ where } \sum_k V_{ijk} = 1, \sum_j T_{ij} = 1$$

3. UNCERTAINTY IN RISK ANALYSIS

Risk analysis must often rely on speculation, best guesses, incomplete data, and many unproven assumptions (The NIST Handbook 1995). Any risk measure based on the scoring method is sensitive to small changes in weights as well as changes in scores. Therefore, the uncertainty issues about scores and weights in the checklist method should be considered. According

to the NIST Handbook (1995), there are two primary sources of uncertainty: (1) a lack of confidence or precision in the risk analysis model or methodology, and (2) a lack of sufficient information to determine the exact value of the elements of the risk model. The correctness of the weight in the checklist method is related to the former while the correctness of the score is associated with the latter. Uncertainty is different from ambiguity; ambiguity is generally handled by the fuzzy set theory in risk analysis. According to Smets' (1991) contrast between imprecision (ambiguity) and uncertainty, imprecision covers cases where the value of a variable is given but not with the required precision, whereas uncertainty covers cases where an agent can construct a personal subjective opinion (belief) in a proposition that is not definitively established. Let us look at the following example: 'How much financial loss is incurred from the disclosure of specific data?' Assume that the analyst is sure that it would be a large loss although he cannot express the exact figure. In this case, the fuzzy theory can be applied. On the contrary, assume he thinks that it could be a large amount but is not sure about this because the actual loss might be much smaller than he expects. This situation represents the uncertainty in the analyst's opinion.

In this article, we will tackle the uncertainties associated with checklist method scores by adopting *plausibility* as a measure of risk, while avoiding the uncertainties associated with the weights by not considering them. In our checklist-based risk analysis, each asset is evaluated from a security preservation perspective by considering all the relevant controls. Non-existence or failure of any control could result in insecurity. This implies that all controls should be regarded as equally important *in terms of the security preservation* (this does not mean that we presume all controls are equally important when assessing the values of controls). With this strategy, we avoid the problem of weight assignment. Plausibility is a term used in belief functions (also known as Dempster-Shafer theory of evidence). Belief function is a general tool for representing someone's degree of belief in an uncertain situation. In this article, plausibility represents the potential insecurity after given evidence of security has been considered.

3.1 Evidential Network

In this article, the overall structure for measuring risk with uncertainty follows the structure of Srivastava's belief function formula (Srivastava and Shafer 1992, Srivastava and Mock 2000). He has developed a special network diagram to apply belief functions to various applications in the accounting domain such as the calculation of audit risk (Srivastava and Shafer 1992) and the WebTrust assurance service (Srivastava and Mock 2000). His model starts from building a network diagram called an evidential

network. In the evidential network, a rounded rectangle represents a variable and a proper rectangle represents evidence, which is connected to a variable that it directly supports. A circle with ‘&’ implies that the variable on the left of the ‘&’ is true if and only if the variables on the right of the ‘&’ are true. Based on his model, we have applied the evidential network to a checklist-based risk analysis, as shown in figure 1. Each variable in an evidential network has a proposition. The proposition at the asset variable is ‘asset *i* is secure’ Similarly, the proposition at the threat variable is ‘threat *j* will not be realised’ and the proposition at the control variable is ‘control *k* has been placed against threat *j*’ It is assumed that the threat will not be realised if all corresponding controls function as intended. However, it does not mean that perfect security can be achieved. Our checklist method corresponds to the baseline approach, which is for baseline protection. The above propositions, ‘asset *i* is secure’ and ‘threat will not be realised’ imply that the risk will be reduced to an acceptable level as specified by the baseline protection. If an organisation feels that the checkpoints currently available for baseline protection are not sufficient to meet its baseline security, it may add some additional checkpoints at its own discretion. The proper rectangle ‘control’ represents the supporting evidence that the control, contributing to the prevention of the threat realisation, has been placed.

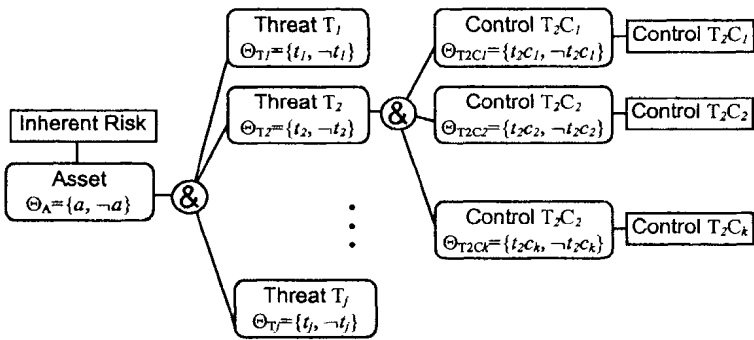


Figure 1. Evidential network for checklist-based risk analysis

This evidential network represents a framework for the checklist method mentioned earlier. The belief in the control corresponds to the score in the checklist method. The degree of belief is a number (not a probability) ranging from 0 to 1. High belief implies that there is strong evidential support for the given proposition. The degree of belief is determined by an analyst’s feeling with respect to the given evidence. For example, the analyst feels that the control seems to be functioning as intended but he is not sure of this for some reason (e.g. he did not perform substantive tests or surveillance tests). He therefore decides to assign a medium level of belief to the

proposition that the control has been placed. At the same time, he has found a control malfunction such as the occasional bypass of the control. This gives him some degree of belief in the negation of the proposition. From these inputs to the 'control' rectangles, the plausibility of threat realisation and the plausibility of overall insecurity are calculated. The rectangle 'inherent risk' represents the potential insecurity resulting from factors beyond the scope of the security review. Examples of such factors are the lack of security awareness, lack of quality security management, potential security flaws and operational mistakes in new systems, and so on. These factors are not listed in the checklist but could cause the insecurity of an asset even if all the controls under review work properly. If the analyst feels that the inherent risk surrounding the asset is high, he will assign a high degree of belief to the negation of the proposition since risk is the opposite concept of security.

3.2 Basic Background for Belief Function Approach¹

3.2.1 *m*-value

In evidence theory, traditional probabilities are replaced by the concept of evidential support. The contrast is between the chance that a hypothesis is true and the chance that the evidence proves that the hypothesis is true (Laskey and Cohen 1986). A frame of discernment, denoted by Θ , represents an exhaustive and mutually exclusive set of possible answers to a question. Instead of using probability, evidence theory uses the function m (called basic probability assignment) that assigns a number $m(B)$ to each subset B of Θ that satisfies:

- a) $m(\emptyset)=0$
- b) $m(B) \geq 0$ for all $B \subseteq \Theta$
- c) $\sum\{m(B) | B \subseteq \Theta\}=1$

The way of assigning m -values in our model is by using a risk analyst's subjective judgment; this is the same as in Srivastava's model. The frame of discernment on the asset variable (Θ_A) has two elements, a and its negation ($\neg a$). Therefore there exist three m -values such as $m_A(\{a\})$, $m_A(\{\neg a\})$ and $m_A(\{a, \neg a\})$. For simplicity, we will write $m_A(a)$ and $m_A(\neg a)$ instead of $m_A(\{a\})$ and $m_A(\{\neg a\})$. The subscript represents the name of variable to which evidence is applied ('A' stands for the asset variable in this case). The

¹ A major part of this section is based on Srivastava and Shafer (1992).

element ‘ a ’ represents, for example, the proposition that ‘asset i is secure’ and the element ‘ $\neg a$ ’ represents the proposition that ‘asset i is NOT secure’. The way of assigning m -values is as follows. Assume that an analyst feels that the given evidence supports the proposition a with a medium level (say, 0.6) of support and he feels that there is no evidence supporting $\neg a$. Thus, he assigns $m_A(a)=0.6$ and $m_A(\neg a)=0$. $m_A(\{a,\neg a\})=1-m_A(a)-m_A(\neg a)$ represents the ignorance (the amount committed to neither a nor $\neg a$).

3.2.2 Belief and Plausibility

The total belief in a subset B of a frame Θ is defined as $Bel(B)=\sum\{m(X)|X\subseteq B\}$ for all $B\subseteq\Theta$, and the plausibility of B is defined as $pl(B)=\sum\{m(X)|B\cap X\neq\emptyset\}=1-Bel(\neg B)$. The value $Bel(B)$ summarises all our reasons for believing B under the given evidence, and the value $pl(B)$ represents how much we should believe B if all currently unknown facts (i.e., underlying ignorance) were to support B . The difference is that $Bel(B)$ quantifies the total amount of justified supports given to B , while $pl(B)$ quantifies the maximum amount of potential supports that could be given to B . Similarly, it can be shown that $pl(\neg B)=1-Bel(B)$, which represents the degree to which $\neg B$ is plausible. In the evidential network in figure 1, every variable has only two propositions and thus the frame on each variable has only two elements. For example, $Bel_X(x)=m_X(x)$ and $Bel_X(\neg x)=m_X(\neg x)$ for a frame $\Theta_X=\{x,\neg x\}$. The plausibility of the negation of the statement has an important interpretation as it represents the measure of risk in our risk analysis model; how plausible is the occurrence of insecurity.

3.2.3 Dempster’s Rule of Combination

If $m_1(B)$ and $m_2(B)$ are two m -values on the same frame Θ induced by two independent evidential resources, then the combined m -value is calculated according to Dempster’s rule (Shafer 1976) which is $m(B)=m_1\oplus m_2(B)=k^{-1}\sum\{m_1(X_1)m_2(X_2)|X_1\cap X_2=B\}$, where $k=1-\sum\{m_1(X_1)m_2(X_2)|X_1\cap X_2=\emptyset\}$, a normalization constant. Normalization is required to satisfy the axiom that the sum of m -values on a frame equals 1 where a conflict ($\sum\{m_1(X_1)m_2(X_2)|X_1\cap X_2=\emptyset\}>0$) exists. Dempster’s rule cannot be used when $k=0$, in which case the two items are not combinable.

3.2.4 Belief Propagation: Forward Direction

In figure 1, the asset is linked with several threats and each threat is linked with several relevant controls. To obtain m -values for the asset variable, we need to calculate the m -values propagated from the input nodes

i.e., controls. The evidential network in figure 1 is an AND-tree, which means that the proposition at the asset variable is true if and only if all the propositions on threat variables are true. Likewise, the proposition at each threat variable is true if and only if all the propositions at the controls associated with the threat are true. First, let us consider the propagation of m -values (from all the controls associated with threat x) to threat x . Assume that there are N controls associated with threat x . Let $m_{T_x C_y}$ ($t_x c_y$), $m_{T_x C_y}(\neg t_x c_y)$ and $m_{T_x C_y}(\{t_x c_y, \neg t_x c_y\})$ be the m -values at control variable $T_x C_y$, obtained from the proper rectangle 'Control $T_x C_y$ '. These m -values are based on an analyst's opinion on the existence/status of control i . The propagated m -values at threat x , denoted by $m_{T_x \leftarrow \text{all } C's \text{ of } T_x}(\theta_{T_x})$ (where $\theta_{T_x} \subseteq \Theta_{T_x}$ and $\theta_{T_x} \neq \emptyset$), are calculated as follows:

1. $m_{T_x \leftarrow \text{all } C's \text{ of } T_x}(t_x) = \prod_{i=1}^N m_{T_x C_i}(t_x c_i)$
2. $m_{T_x \leftarrow \text{all } C's \text{ of } T_x}(\neg t_x) = 1 - \prod_{i=1}^N [1 - m_{T_x C_i}(\neg t_x c_i)]$
3. $m_{T_x \leftarrow \text{all } C's \text{ of } T_x}(\{t_x, \neg t_x\}) = 1 - m_{T_x \leftarrow \text{all } C's \text{ of } T_x}(t_x) - m_{T_x \leftarrow \text{all } C's \text{ of } T_x}(\neg t_x)$

The propagation from the threat variables to the asset variable is similar to the above. Let P be the number of threats. The propagated m -values at the asset variable, denoted by $m_{A \leftarrow \text{all } T's}(\theta_A)$ (where $\theta_A \subseteq \Theta_A$ and $\theta_A \neq \emptyset$), are as follows:

4. $m_{A \leftarrow \text{all } T's}(a) = \prod_{i=1}^P m_{T_i \leftarrow \text{all } C's \text{ of } T_i}(t_i)$
5. $m_{A \leftarrow \text{all } T's}(\neg a) = 1 - \prod_{i=1}^P [1 - m_{T_i \leftarrow \text{all } C's \text{ of } T_i}(\neg t_i)]$
6. $m_{A \leftarrow \text{all } T's}(\{a, \neg a\}) = 1 - m_{A \leftarrow \text{all } T's}(a) - m_{A \leftarrow \text{all } T's}(\neg a)$

3.2.5 Measure of Risk

From equations 1, 2 and 3, we can obtain the m -values propagated from the controls to the threats. These m -values are also propagated to the asset variable and the results of this propagation are obtained from equations 4, 5 and 6. We apply Dempster's rule to combine these propagated m -values with the m -values (denoted by $m_A(\theta_A)$, where $\theta_A \subseteq \Theta_A$ and $\theta_A \neq \emptyset$) obtained from the proper rectangle 'inherent risk'. This yields the following m -values for the asset variable:

7. $m_A^t(a) = m_A \oplus m_{A \leftarrow \text{all } T's}(a)$
8. $m_A^t(\neg a) = m_A \oplus m_{A \leftarrow \text{all } T's}(\neg a)$
9. $m_A^t(\{a, \neg a\}) = 1 - m_A^t(a) - m_A^t(\neg a)$

The superscript ' t ' indicates that these m -values are the resulting (total) m -values after all evidence in the AND-tree has been considered. As

mentioned above, the plausibility of the negation of the proposition at the asset variable, $pl_A(-a)=1-Bel_A(a)$, is the measure of risk in our model, which represents the degree to which insecurity is plausible. The measure of risk, $pl_A(-a)$ is $1-m^t_A(a)$ as $Bel_A(a)=m^t_A(a)$.

3.2.6 Belief Propagation: Backward Direction

The belief (m -values) is also propagated in the opposite direction (from asset to each threat, and from each threat to relevant controls) as well as the propagation mentioned above. We need to consider this propagation to obtain the resulting m -values for threats and controls. It means that the m -values obtained from the inherent risk node should be propagated to each threat and to each control since this inherent risk affects the security status at threat and control levels. We need to consider this propagation when we want to obtain the marginal risk measures such as the plausibility of the threat realisation and the plausibility of the control malfunction/failure. These measures are not required for the calculation of the measure of risk at asset level. However, it provides useful information to management (e.g. which threat is likely to be realised and which control is likely not to be guaranteed). The m -values propagated from the asset to threat x , denoted by $m_{T_x \leftarrow A}$ & all other T 's (θ_{T_x}) (where $\theta_{T_x} \subseteq \Theta_{T_x}$ and $\theta_{T_x} \neq \emptyset$), are as follows:

- 10. $m_{T_x \leftarrow A}$ & all other T 's (t_x) = $k_x^{-1} m_A(a) \prod_{i=1, i \neq x}^P [1 - m_{T_i \leftarrow \text{all } C\text{'s of } T_i}(-t_i)]$
- 11. $m_{T_x \leftarrow A}$ & all other T 's ($-t_x$) = $k_x^{-1} m_A(-a) \prod_{i=1, i \neq x}^P m_{T_i \leftarrow \text{all } C\text{'s of } T_i}(t_i)$
- 12. $m_{T_x \leftarrow A}$ & all other T 's ($\{t_x, -t_x\}$) = $1 - m_{T_x \leftarrow A}$ & all other T 's (t_x) - $m_{T_x \leftarrow A}$ & all other T 's ($-t_x$)

where k_x is the normalization constant, which is given by $k_x = 1 - m_A(a) \cdot C_x$, where C_x is given by $C_x = 1 - \prod_{i=1, i \neq x}^P [1 - m_{T_i \leftarrow \text{all } C\text{'s of } T_i}(-t_i)]$

Then, the resulting m -values at threat x are as follows:

- 13. $m^t_{T_x}(t_x) = m_{T_x \leftarrow \text{all } C\text{'s of } T_x} \oplus m_{T_x \leftarrow A}$ & all other T 's (t_x)
- 14. $m^t_{T_x}(-t_x) = m_{T_x \leftarrow \text{all } C\text{'s of } T_x} \oplus m_{T_x \leftarrow A}$ & all other T 's ($-t_x$)
- 15. $m^t_{T_x}(\{t_x, -t_x\}) = 1 - m^t_{T_x}(t_x) - m^t_{T_x}(-t_x)$

Similarly, the m -values propagated from threat x to control y (that is associated with threat x), denoted by $m_{T_x C_y \leftarrow T_x}$ & all other C 's of T_x ($\theta_{T_x C_y}$) where $\theta_{T_x C_y} \subseteq \Theta_{T_x C_y}$ and $\theta_{T_x C_y} \neq \emptyset$), are as follows:

- 16. $m_{T_x C_y \leftarrow T_x}$ & all other C 's of T_x ($t_x C_y$) = $k_{xy}^{-1} m_{T_x \leftarrow A}$ & all other T 's (t_x) $\prod_{i=1, i \neq y}^N [1 - m_{T_x C_i}(-t_x C_i)]$
- 17. $m_{T_x C_y \leftarrow T_x}$ & all other C 's of T_x ($-t_x C_y$)

$$=k_{xy}^{-1}m_{Tx \leftarrow A} \text{ \& all other T's } (-t_x) \prod_{i=1, i \neq y}^N m_{TxCi}(t_x c_i)$$

18. $m_{TxCy \leftarrow Tx} \text{ \& all other C's of Tx } (\{t_x c_y, \neg t_x c_y\})$

$$=1-m_{TxCy \leftarrow Tx} \text{ \& all other C's of Tx } (t_x c_y) - m_{TxCy \leftarrow Tx} \text{ \& all other C's of Tx } (\neg t_x c_y)$$

where k_{xy} the normalization constant, which is given by $k_{xy}=1-m_{Tx \leftarrow A} \text{ \& all other T's } (t_x) \cdot C_{xy}$, where C_{xy} is given by $C_{xy}=1-\prod_{i=1, i \neq y}^N [1-m_{TxCi}(\neg t_x c_i)]$

Then, the resulting m -values at control y that is associated with threat x are as follows:

19. $m'_{TxCy}(t_x c_y) = m_{TxCy} \oplus m_{TxCy \leftarrow Tx} \text{ \& all other C's of Tx } (t_x c_y)$

20. $m'_{TxCy}(\neg t_x c_y) = m_{TxCy} \oplus m_{TxCy \leftarrow Tx} \text{ \& all other C's of Tx } (\neg t_x c_y)$

21. $m'_{TxCy}(\{t_x c_y, \neg t_x c_y\}) = 1 - m'_{TxCy}(t_x c_y) - m'_{TxCy}(\neg t_x c_y)$

4. EXAMPLES

4.1 Security of Data Asset

In this section, we provide a numerical example to show how our risk analysis method can be used. Our example is the security review of the controls for securing a specific data asset. The review of security preservation on a data asset requires an extensive review process because the security of data asset could be affected by many sources of insecurity. For example, an attacker could get access to the data asset by exploiting security flaws in the operating system. However, the full simultaneous examination of all the possible security holes that could affect the security of the data asset is not efficient in the checklist method since it generates a very lengthy list of questions for each asset. Therefore, some assumptions and omissions are required to achieve quick and efficient reviews. The main purpose of the baseline approach is to ensure that all identified assets are protected to a baseline level. Once we assume that all the major vulnerabilities are identified by relevant baseline security reviews, we can narrow down the focus of our review. The review of the data asset in our example assumes that other vulnerabilities have been examined in other review categories. For example, unavailability of the data asset has been excluded as it is assumed that this issue is to be examined by the reviews for unavailability of server, client, network component and backup media. For simplicity, our example includes only two threats with three controls per threat. The overall structure of the example is shown in figure 2.

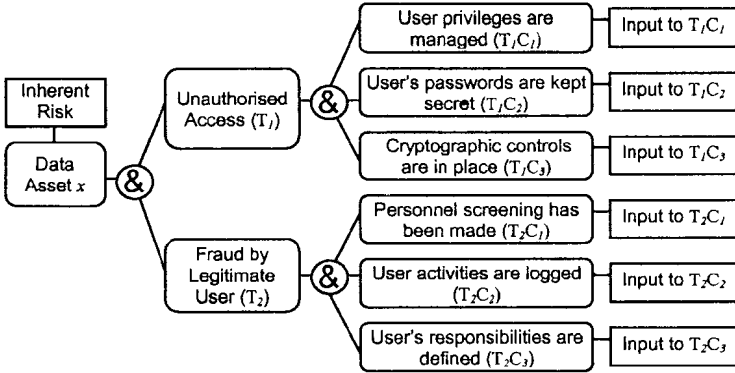


Figure 2. The example network for data asset checklist

Assume that a risk analyst has reviewed the client application for the data asset x against relevant controls, and provided the m -values for each control as shown in table 1. The way of assigning m -values can be illustrated with the following example. Consider that the analyst has found that well-defined access control lists exist in the organisation but he is not sure whether they preserve the principle of least privilege. Therefore, he assigned 0.6 as support for the proposition that user privileges are managed. At the same time, he assigned 0 as support for the negation of the proposition since he did not find any evidence of bad user privilege management practices. These values are shown in the first column ($T1C1$) in table 1.

Table 1. Input Values for Example

	$T1C1$	$T1C2$	$T1C3$	$T2C1$	$T2C2$	$T2C3$	Inherent Risk
support	.6	.5	.7	.9	.7	.7	.7
Neg. Support	.0	.3	.2	.0	.2	.0	.0

As for the input values for inherent risk, the analyst feels that there is top management commitment to security and a mature security culture in the organisation; these are positive factors for the proposition that the data asset is protected from the viewpoint of user's behaviour. He thus assigned 0.7 support for the proposition that the data asset x is protected. Excluding the last column, the values in table 1 represent the m -values for $m_{T1Cj}(t1c_j)$ and $m_{T1Cj}(\neg t1c_j)$ ($i=1,2$ and $j=1,2,3$). The values in the last column represent the m -values for $m_A(a)$ and $m_A(\neg a)$. Based on these m -values provided by the analyst, we can calculate all the m -values required for obtaining the measure of risk; how plausible the insecurity of the data asset x is. The calculation results are shown in table 2.

Table 2. Calculation Procedures for Measure of Risk

$m_{T1 \leftarrow \text{all } C's \text{ of } T1}(t_1) = 0.210$, $m_{T1 \leftarrow \text{all } C's \text{ of } T1}(\neg t_1) = 0.440$
$m_{T2 \leftarrow \text{all } C's \text{ of } T2}(t_2) = 0.441$, $m_{T2 \leftarrow \text{all } C's \text{ of } T2}(\neg t_2) = 0.200$
$m_{A \leftarrow \text{all } T's}(a) = 0.093$, $m_{A \leftarrow \text{all } T's}(\neg a) = 0.552$
$m'_A(a) = 0.556$, $m'_A(\neg a) = 0.270$ $\therefore pl_A(\neg a) = 0.444$

$m_{T1 \leftarrow A \text{ \& all other } T's}(t_1) = 0.651$, $m_{T1 \leftarrow A \text{ \& all other } T's}(\neg t_1) = 0.000$
$m'_{T1}(t_1) = 0.614$, $m'_{T1}(\neg t_1) = 0.215$ $\therefore pl_{T1}(\neg t_1) = 0.386$
$m_{T2 \leftarrow A \text{ \& all other } T's}(t_2) = 0.566$, $m_{T2 \leftarrow A \text{ \& all other } T's}(\neg t_2) = 0.000$
$m'_{T2}(t_2) = 0.727$, $m'_{T2}(\neg t_2) = 0.098$ $\therefore pl_{T2}(\neg t_2) = 0.273$

$m_{T1C1 \leftarrow T1 \text{ \& all other } C's \text{ of } T1}(t_1c_1) = 0.511$, $m_{T1C1 \leftarrow T1 \text{ \& all other } C's \text{ of } T1}(\neg t_1c_1) = 0.000$
$m_{T1C2 \leftarrow T1 \text{ \& all other } C's \text{ of } T1}(t_1c_2) = 0.599$, $m_{T1C2 \leftarrow T1 \text{ \& all other } C's \text{ of } T1}(\neg t_1c_2) = 0.000$
$m_{T1C3 \leftarrow T1 \text{ \& all other } C's \text{ of } T1}(t_1c_3) = 0.566$, $m_{T1C3 \leftarrow T1 \text{ \& all other } C's \text{ of } T1}(\neg t_1c_3) = 0.000$
$m'_{T1C1}(t_1c_1) = 0.804$, $m'_{T1C1}(\neg t_1c_1) = 0.000$ $\therefore pl_{T1C1}(\neg t_1c_1) = 0.196$
$m'_{T1C2}(t_1c_2) = 0.756$, $m'_{T1C2}(\neg t_1c_2) = 0.147$ $\therefore pl_{T1C2}(\neg t_1c_2) = 0.244$
$m'_{T1C3}(t_1c_3) = 0.853$, $m'_{T1C3}(\neg t_1c_3) = 0.098$ $\therefore pl_{T1C3}(\neg t_1c_3) = 0.147$

$m_{T2C1 \leftarrow T2 \text{ \& all other } C's \text{ of } T2}(t_2c_1) = 0.511$, $m_{T2C1 \leftarrow T2 \text{ \& all other } C's \text{ of } T2}(\neg t_2c_1) = 0.000$
$m_{T2C2 \leftarrow T2 \text{ \& all other } C's \text{ of } T2}(t_2c_2) = 0.566$, $m_{T2C2 \leftarrow T2 \text{ \& all other } C's \text{ of } T2}(\neg t_2c_2) = 0.000$
$m_{T2C3 \leftarrow T2 \text{ \& all other } C's \text{ of } T2}(t_2c_3) = 0.511$, $m_{T2C3 \leftarrow T2 \text{ \& all other } C's \text{ of } T2}(\neg t_2c_3) = 0.000$
$m'_{T2C1}(t_2c_1) = 0.951$, $m'_{T2C1}(\neg t_2c_1) = 0.000$ $\therefore pl_{T2C1}(\neg t_2c_1) = 0.049$
$m'_{T2C2}(t_2c_2) = 0.853$, $m'_{T2C2}(\neg t_2c_2) = 0.098$ $\therefore pl_{T2C2}(\neg t_2c_2) = 0.147$
$m'_{T2C3}(t_2c_3) = 0.853$, $m'_{T2C3}(\neg t_2c_3) = 0.000$ $\therefore pl_{T2C3}(\neg t_2c_3) = 0.147$

The measure of risk in this example is 0.444. Table 2 also shows the final m -values at threat and control variables after belief propagation in the backward direction. The plausibility of the realisation of unauthorised access is 0.386 whereas the plausibility of the realisation of fraud attempt is 0.273. From these results, we can conclude that unauthorised access is more likely to occur than a fraud attempt. As our evidential network diagram indicates, the measure of risk is sensitive to the changes of m -values arising from the inherent risk node. If the analyst is competent and has enough knowledge of the organisation's information system, consideration of the inherent risk would provide a more precise reflection of organisational security issues. Otherwise, it may produce the wrong result. Therefore, if he cannot provide any opinion/answer to the question X in the checklist, he may leave the question unanswered. In this case, $m_x(x) = m_x(\neg x) = 0$ will be assigned, which means $m_x(\{x, \neg x\}) = 1$. This is equally applicable to the inherent risk node. Another concern in using evidential reasoning is that this approach requires more inputs than conventional checklist methods. If we assume that the evidence is affirmative, i.e., the evidence supports a proposition and does not support its negation, the number of inputs required may be reduced to the levels found in the conventional checklist method. Obtaining risk measures by evidential reasoning includes a tedious calculation process. Therefore, a computerised facility for belief calculation should be embedded in the checklist-based analysis tool.

4.2 BS7799 Self-Assessment by Evidential Reasoning

The evidential network specified in this article can also be used for self-assessment of BS7799 compliance without any major modification of the network structure. Self-assessment refers to the assessment performed by an organisation (internally) to check whether it is ready for a formal assessment against the Accredited Certification Scheme for BS7799 Part 2. BS7799 Part 2 (1999) provides a summarised list of controls and control objectives in ten assessment categories so that the controls specified by BS7799 can be examined more clearly.

To apply BS7799 self-assessment, the asset variable in figure 1 should be replaced by the variable 'assessment category', the threat variable should be replaced by the variable 'control objective', and the control variable should be replaced by the variable 'control procedure' that ensures the relevant control objective. For example, BS7799 Part 2 specifies three control objectives for physical and environmental security (Assessment Category 5). One is to prevent unauthorised access, damage and interference to business premises and information (Control Objective 5.1). Another is to prevent loss, damage or compromise of assets and interruption to business activities (Control Objective 5.2). The third is to prevent compromise or theft of information and information processing facilities (Control Objective 5.3). Each control objective is associated with several relevant controls. For example, there are two controls for the Control Objective 5.3:

- Clear desk and clear screen policy: Organisations shall have and implement a clear desk and a clear screen policy in order to reduce the risks of unauthorised access, loss, and damage to information (Control 5.3.1)
- Removal of property: Equipment, information or software belonging to the organisation shall not be removed without authorisation (Control 5.3.2).

The inherent risk node in figure 1 can also be used in the BS7799 self-assessment structure since the concepts are still valid. The proposition at assessment category x is that category x satisfies the certification requirements. The proposition at the control objective $x.y$ is that the control objective y belonging to the category x meets the requirements for the certification. The proposition at the control $x.y.z$ is that the control procedure z belonging to the control objective $x.y$ satisfies the control requirement for the certification. The guidelines for assessing each control against BS7799 certification are provided in 'BS7799: Preparing for BS7799 certification (1999)'.

5. CONCLUSION

In this article, we have examined the applicability of evidential reasoning in the risk analysis domain. Most risk analysis methods force the analyst to provide subjective opinions in the course of the evaluation. Therefore, the belief function approach can be an alternative to these conventional risk analysis methods. Because evidential reasoning using AND-trees originated from accounting domains such as audit risk assessment, it could be modified to accommodate checklist-based risk analysis without much difficulty. Belief functions can be used instead of the conventional approach based on probability theory. The advantage of using belief functions is their ability to deal with uncertainty. In probability theory, the sum of the probability of event occurrence and the probability of its complement should be 1. However, this restriction is relaxed in evidence theory by introducing the concept of ignorance.

The major drawback of quantitative risk analysis methodologies is the difficulty in estimating probabilities since quantitative methods rely heavily on the accuracy of the estimates. Although evidence theory does not provide a clear answer to this problem (since it still requires m -values, which are regarded as meta probabilities over a probability that is to be estimated), it may provide, to some extent, the relaxation in accuracy needed when expressing uncertainty. The problem of qualitative risk analysis is that the risk is often based on subjective judgment. Although detailed guidelines for assigning qualitative values are provided in many qualitative methods, the answers provided by analysts still rely on their own subjective opinions. This problem can also be handled by evidence theory. In summary, evidence theory offers a new way of thinking about risk analysis.

REFERENCES

- BS7799: Guide to BS7799 Risk Assessment and Risk Management, British Standard Institution, 1998
- BS7799: Part 1: Code of Practice for Information Security Management, British Standard Institution, 1999
- BS7799: Part 2: Specification for Information Security Management Systems, British Standard Institution, 1999
- BS7799: Preparing for BS7799 Certification, British Standard Institution, 1999
- CCTA, An Overview of CRAMM, CCTA IT Security and Privacy Group, UK, 1990
- Ciechanowicz, Z., Risk Analysis Requirements, Conflicts and Problems, Computers & Security, 1997, Vol. 16, No. 3, pp. 223-232
- Erwin, D., The Thirty-Minute Risk Analysis, Information Systems Security, 1994, Vol.1.3, No. 3, pp. 37-44

- GISA (German Information Security Agency), IT-Baseline Protection Manual, Bundesamt für Sicherheit in der Informationstechnik, 1997
- ISO/IEC TR 13335-1, Guideline for Management of IT Security-Part1: Concepts and Models for IT security, 1996
- ISO/IEC TR 13335-2, Guideline for Management of IT Security-Part2: Managing and Planning of IT Security, 1997
- ISO/IEC TR 13335-3, Guideline for Management of IT Security-Part3: Techniques for the Management of IT Security, Working Draft, 1997
- ISO/IEC TR 13335-4, Guideline for Management of IT Security-Part4: Selection of Safeguards, 1999
- Laskey, K.B. and Cohen, M.S., Applications of the Dempster-Shafer Theory of Evidence, Proceedings of the 1986 Winter Simulation Conference, December 8-10, 1986, Washington, DC, pp. 440-444
- NIST, An Introduction to Computer Security: The NIST Handbook, Special Publication 800-12, 1995
- Pfleeger, C.P., Security in Computing, 2nd Edition, Prentice-Hall, NJ, 1997
- Rainer, R.K., Snyder, C.A. and Carr, H.H., Risk Analysis for Information Technology, Journal of Management Information Systems, 1991, Vol. 8, No. 1, pp. 129-147
- Shafer, G.R., A Mathematical Theory of Evidence, Princeton University Press, NJ, 1976
- Smets, P., Varieties of Ignorance and the Need for Well-Founded Theories, Information Sciences, 1991, Vol. 57-58, pp. 135-144.
- Srivastava, R.P. and Mock, T.J., Evidential Reasoning for WebTrust Assurance Services, Journal of Management Information Systems, 1999-2000; Vol. 16, No. 3, pp. 11-32
- Srivastava, R.P. and Shafer, G.R., Belief-Function Formulas for Audit Risk, The Accounting Review, 1992, Vol. 67, No. 2, pp. 249-283

Improving the Protection of Assets in Open Distributed Systems by Use of X-ifying Risk Analysis

Ann FRISINGER

The Royal Institute of Technology

Department of Teleinformatics

Email: afri@it.kth.se

Phone: +46-8-7931321

Mobile: +46-70-7931321

Key words: *Information security, data security, open distributed system, risk analysis, X-ifying factors.*

Abstract: *Open distributed systems operate in a networked global space where parts are owned by - thus can be controlled by - the local system owner, but most parts are shared globally. The system boundaries are fuzzy and we can only count the system owner to control his/her assets at some point of time. The generic system parts - those shared globally - are data oriented. The specific system parts - those owned by the local system owner - are information oriented. This fact should have impact on the way we view assets when setting the right security requirements. Many approaches focus only on the generic parts, i.e. to protect the data. They thereby overlook the informational aspect of the asset, In order to find these specific requirements, it is important to analyze the risks related to information so that it can be protected in a satisfactory way. This paper will describe how the problem can be solved by use of a risk analysis approach with so called X-ification. X-ifying is a way of mating together the best available experience with values appropriate for the local target system X. X-ifying factors differ depending on if you look at assets from a data or informational point of view. They also change in importance from asset to asset, from industry to industry and from person to person.*

1. Introduction

Creating security and mutual trust in an open distributed environment, like the Internet, where exchange of data is an emerging phenomenon are essential but difficult to establish. Open distributed systems operate in a networked global space where parts are owned by - thus can be controlled by - the local system owner, but most parts are shared globally. The system boundaries are fuzzy and we can only count the local system owner to control his/her assets at some point of time. The generic system parts - those shared globally where the local system owner does not have control and where assets float around as uninterpreted bits and bytes - are data oriented. The specific system parts - those owned by the local system owner and where s/he has control and is in charge of media where the asset resides - are information oriented. This fact should have impact on the way we view assets when setting the right requirements for security.

A common view in industry is that you can buy a black box with technology which solves all your problems with security. "The majority of defense dollars go for essential functions such as firewalls" [FOR98, p 4]. Many efforts aiming to protect assets in open distributed systems focus on the generic parts, i.e. to protect the data. This is valid for baseline security approaches such as the British Standard 7799, BS7799, Code of Practice for Information Security Management [BS7799, SIS99], and also for traditional first generation risk analysis methods which base the risk assessment on statistics from before and then focus on selecting countermeasures from an existing limited list of solutions [BAS93, HAM96]. They thereby set a basic level of security for the data. However they overlook the informational aspect of the asset, i.e. where a piece of data is interpreted by a human being, i.e. put into context, and turned into information that may need specific protection. In order to find these specific requirements it is important to analyze the risks related to information so that it can be protected in a satisfactory way, considering also the costs that would entail.

The requirements for security differ from asset to asset, from industry to industry [FRI00b] and from person to person [JON99]. We call the variations X-ifying factors [FRI00b]. X-ifying factors could be wise to consider in a risk analysis. We can see that X-ifying factors also differ depending on if you look at assets from a data or informational point of view.

This paper will discuss the dilemma that asset's informational values are often overlooked when deciding the requirements for protection in an open distributed system. It will describe an approach to overcome that problem by utilizing risk analysis with so called X-ification. Furthermore

it will describe how these X-ifying factors differ from industry to industry and from person to person and how this impact the view of the asset values.

2. Information needs specific protection, data needs generic

System theory is a scaleable concept that gives a possibility to integrate delimited parts into a whole system, where sub-systems play together with relations and interactions. This is a base for a holistic perspective. A holistic approach is recommended as an instrument to structure the sub-systems into an integrated whole before the final solution is put into reality. This is to avoid that piecemeal security solutions are applied to isolated problem areas where no consideration is given to the integration [YNG96, FIL99].

In efforts to control, humans may choose to either assume that the reality is a system or could be looked upon as a system through the learning process i.e. humans can learn how the concept of a system reflects the real world. The control method in the first case could be labeled systematic engineering or hard systems thinking. The second one systemic or soft system thinking.

In soft systems thinking (the epistemological science) where we explore intangible areas that are difficult to capture, perceived realities are treated as problems and are solved in a systemic way. Soft systems thinking focus on the need to synthesize and to use induction to create value. It encourages an open living approach to open dynamic research objects [YNG96, MAG99]. This gives a possibility to delimit into subsystems, and to control when to move from a soft systems approach to a hard system approach going systematic. In addition to the holistic perspective a systemic approach is needed. When combining the systemic with the holistic view, the approach brings some useful principles [YNG96, p27]:

- Delimit the system of study from the environment,
- Define the existing environment,
- Define the inflow, throughflow, and outflow, and
- Structure the built-in control system so that it can deal with inner and outer variety,

In hard systems thinking (the ontological science) perceived realities are treated as existing systems and their problems are solved by systematic methods [YNG96, MAG99]. Deduction and proofs based on

exact rules of procedures are foundation for ontological science. A systematic or synonymously methodical approach can be defined as an orderly, regular procedure for obtaining an object.

[FIL99, p199] says “Information security is contextual in nature, and difficult to view as a system in itself” and Information security is contextual and not systemic” [FIL99, p187]. We agree that information security is contextual but as opposite to [FIL99] we see that open distributed emergent systems are well suited to be approached by soft systems thinking, where the meaning of the data is related to a context thus forming information.

The systemic view enables us to focus on our assets as information where we can control them and set specific security requirements for them. When dealing with parts that we do not control, we can still keep our requirements for assets that were decided when we took the informational view, but we have no control that the owner of the media where asset travels/resides will meet our desired level of protection, unless we find a way to totally encapsulate the asset. Similarly we will in our system host assets that are owned by others, thus we see those assets as data or we make our own interpretation of those asset’s values, however those might not agree with the owner’s wish, unless very clearly specified. To be able to set a basic level of security for hosted assets etc. it is recommended to take a systematic approach (hard systems thinking). Perhaps take help from a baseline security approaches such as the BS7799.

Soft systems and hard systems thinking are complementary to each other. The systematic and methodical view represents reality, stability, and consistency while the systemic and amethodical [TRU00] view represents interpretation, change and encourages innovation that leads to adaptation.

While a systematic approach would encourage the risk analyst to find patterns from the past to build future trends for incident risks and costs, the systemic and amethodical approach would guide us to adapt the best available experience.

The systematic approach is used in traditional approaches to security and usually produces solutions for data. When combined with a systemic and amethodical approach it is possible to also focus on information, thus produce specific solutions for assets.

3. How risk analysis with X-ification adds value

X-ifying is a way of mating together the best available experience with data appropriate for the target system X [FRI99]. System X is an open distributed system and X is the first system in its generation. X' is the successor of system X. The customizing data can be an assessment of asset attractiveness, actor capabilities, system and media vulnerabilities, and business values. By utilizing X-ification capabilities, we can compensate for errors and disbelief in estimates, non-existing or outdated experience numbers. The X-ification considers the meaning of the data, i.e. information, and adapts the best available previous experience (statistics) to the local application area.

A risk, R, can be calculated as the product of probability, P, and cost, C according to formula 1 [FRI00a]:

$$(1) R = P * C$$

The probability P and the cost C can each be compiled as the product of various elements, according to formula 2 and 3:

$$(2) P = \{(Capability_of_actor * Attractiveness_of_asset) * Vulnerability_in_media_where_asset_resides\} * Method_of_operation_used_to_force_the_system * Objective_of_actor$$

$$(3) C = (Direct_loss_value * Cost_to_recover_repair) * Indirect_cost$$

The various parameters and operations in the formulas can in a more or less extent be utilized when performing X-ification in a risk analysis. The range of the X-ifying effect depends much on the usage. If we for instance decide to use assessed values when calculating the first part of probability for attack, i.e. $\{(Capability_of_actor * Attractiveness_of_asset) * Vulnerability_in_media_where_asset_resides\}$, and if we then could find values for the second part, i.e. $\{Method_of_operation_used_to_force_the_system * Objective_of_actor\}$, that are based on prior experience, those could be used as an index key where the index sums up to 1 (i.e. what is the distribution if hit by an incident). With help of the index, linguistic variables and fuzzy set theory, we can transform/calibrate the assessed probability values into new. There are also other ways of consolidating risk/probability values into one. In [VEN99] the consolidation of the risk values was performed by calculating the average of risk values. The way you do it, the importance and weight you give to different parameters, will impact the result and the level of X-ification.

This method of splitting a variable into two, one representing an assessed value and one representing a statistical value (or best available experience from the past), could be done for several elements in the formula, depending on availability of values. We call it X-ification.

4. X-ifying factors' dependency on industry and person

X-ifying factors are useful in risk analysis, especially when considering information oriented aspects. The factors differ from industry to industry and from person to person.

Different sectors and application areas have specific features that would make it possible to use a generic model for risk analysis but apply an X-ified factor. This requires the analyst to know what differs a specific system from an average system. [FRI00b] looked at high level factors with influence on security which were labeled X-ifying factors. The research, performed in form of a survey, showed there exist characteristic factors that differentiate between industry sectors. For instance, there are more hackers in university environment, the security awareness is generally higher in bank and finance sector, and the result of an incidence within the healthcare sector could be disastrous with impact on human lives, etc. Table 1 shows a summary of assessed values of X-ifying factors/parameters for the selected industry sectors of bank&finance, education, government, healthcare, IT&telecom, manufacturing, sales&distribution. The assessed values, based on answers in the survey, for an industry sector relative the average system and are given by linguistic variables {l=low, m=medium, h=high}.

These characteristic areas could be translated into a weight in the risk analysis algorithm in the following way, for example propose that for a bank the direct cost should be weighted higher in the algorithm and indirect cost would be weighted much higher than the average system, while vulnerability level is lower due to high security awareness and well developed systems. This varies then from a university where the direct cost and indirect cost should be weighted a bit less than the average system while the vulnerability level should be weighted higher than average, due to low security awareness and a lot of hackers playing with the system. The information could be utilized as a source of ideas when X-ifying in a risk analysis and when bridging values from one industry to another.

X-ifying parameter	Bank & Finance	Educ.	Gover nment	Health care	IT & Telco	Manufact	S&D
Capability_of actor	m	h	h	l	m	l-m	l
Attractiveness_of_asset	h	l-m	h	l-m	m	m	m
Vulnerability_in_media_where_asset_resides	l	h	l-m	m-h	m	m	m-h
Direct_loss_value	h	l	m-h	h	m-h	l-m	l
Cost_to_recover_repair	m-h	l	m-h	h	m	l	l
Indirect_cost	h	l	h	h	m	m	m-h

Table 1: X-ifying parameters for industry sectors

A work to provide an understanding, verification, measurement, and prioritization of users’ security requirements in a networked education system was performed in form of a survey, a web based questionnaire, divided into three parts investigating confidentiality, integrity, and availability aspects. The survey was targeting different selected user groups; teachers, students, system administrators. Each part of the survey included a number of scenarios where each scenario related to an asset (information) and which could afterwards be translated into a security requirement for that specific asset [JON99]. The study showed that users have high expectations on security in the selected system. Integrity of the assets was estimated very high, whereas availability of systems and assets was rated almost as important as integrity, and the confidentiality aspect was also valued quite high.

The [JON99] study also showed that different users value assets differently depending on how they are related to them. If the user can put the asset/data into context it is valued higher than if the opposite shows. If the asset is seen as data only, the user only requires it to have some generic protection. X-ification can in this case add a contextual view of data as well as the opposite, it can add a generic view of information. Thus it balances the two views and ensures both aspects are considered before a solution is selected and implemented.

5. Practical experience of using X-ification

A generic method for performing security evaluations of open distributed systems was developed in [FRI99]. The method was generic enough to handle all types of systems, real as well as virtual (i.e. emergent systems or system in a development stage). It enabled successive evaluations, invited feedback from the past and adjusted systems over time. One of the novel features of the method was the X-ifying capability.

The method was tested on a networked reference system in the education sector, but the method is general enough to handle other types of systems, for instance X. The education system, called NED, was evaluated twice, first in its virtual version, then in its real existence, i.e. the first generation of its kind. In the test of the NED, the general evaluation method was adjusted by "NED-ifying" our criteria. The NED-ifying capability introduced both testability and context into results. The method integrates thereby whatever objective and subjective data available with the aim of making the best possible decision within limited time and other resource constraints

During the first evaluation, NED was a virtual system and we did not have any NED incident related data but used the best available statistics from another system as an index thus blending it together with our own assessments for probabilities. The second evaluation used recently collected NED incident related data and we used that in a similar way, i.e. as an index melting it together with assessments. The second evaluation suggested to collect new incident data and limit the time span for the index key to 6-12 months in order to let the experience base reflect recent reality. In both cases the number of incidents used as experience was small. The experience from collecting incident related numbers from the NED system, supported also by previous studies like [BAS93] and [HOW97], indicates that seemingly low incident numbers is a fact to live with (during a six months period we managed to collect 12 incidents). This is especially true if the time for collection is short. We find the number of incidents sufficient to use as check and balance in the NEDification provided reevaluations are repeated on a regular basis. If not done so often, but still on a regular basis, the incident numbers will increase, but then the experience will not be recent and a risk is that the system has evolved much since starting collecting data. A suggestion would then be to balance experience collected from systems, say X+ X'+ X" with recent data from X" and apply it for X".

We used the knowledge of users' security requirements when forecasting costs. We divided the cost into a direct cost and an indirect cost. Furthermore the direct cost was split into a direct loss value and a cost to recover and to repair. We let the opinion of the owner of assets (teachers) reflect the direct loss values, we let the opinion of system administrators reflect the cost to recover and to repair, and lastly we let the end-users (students) decide the indirect cost values. We believe this approach is very suitable to use in environments where the costs or other impacts by tradition has not been estimated before (e.g. education sector) or where consequences are difficult to quantify due to their non-financial nature (e.g. health-care sector), although the approach is generic and could be useful in all sectors.

All in all we experienced that the integration of whatever objective and subjective data available was practical and helped us make the best possible decision considering both the data as well as the informational aspect of assets.

The method enabled the system to adjust to current reality over time where protecting system measures that are hit by many incidents could be replaced by new generations of protecting measures (this is excluding the event of system hit by a very severe incident with a high-level of damage, which will have impact on most of the protecting measures). The protecting measures that are best suited to solve the problem will be promoted to be used in the next version of the system.

7. Conclusions

A starting and essential point for creating security and mutual trust in an open distributed environment is to set the security requirements correctly for assets from the start and then monitor changes regularly.

In this paper we have discussed the importance of addressing both data and informational aspects of assets when setting the security requirements and how this can be achieved by use of a risk analysis with so called X-ification capability. Table 2 shows a brief comparison between data and information.

X-ifying factors differ depending on if you look at assets from a data or informational point of view. They also change in importance from asset to asset, from industry to industry and from person to person. It is thus important to be aware of the differences between how assets are viewed, as data or information. These differences should then be considered in the risk analysis. The outcome of a risk analysis which uses

X-ification techniques will be a prioritized list of security requirements which will balance a data and informational aspect of assets.

Data	Information
Data is uninterpreted bits and bytes	Information is data that is interpreted by someone and put into context
Data has generic value	Information has specific value varying from person to person, and from industry to industry
A methodical approach can be sufficient to decide generic security requirements with focus on media where data resides, e.g. baseline security or early generation risk analysis methods	Methodical approach with amethodical elements is required to decide specific security requirements with focus on the meaning of the data, e.g. risk analysis with X-ification
The media owner implements generic and specific protection based on requirements	The information owner sets specific requirements for protection

Table 2: A comparison between data and information

References

- [BAS93] Richard Baskerville, 'Information Systems Security Design Methods: Implications for Information Systems Development', 1993, ACM Computing Surveys, Vol.25, No.4, pp.375-414.
- [BS7799] BS7799: Code of Practice for Information Security Management, British Standards Institute 1995
- [FIL99] Helen Leslie Fillery-James, 'A Soft Approach To Management of Information Security', July 1999, Doctoral thesis, Curtin University of Technology.
- [FRI99] Ann Frisinger, Louise Yngström 'An approach to standardizing security analysis methods for virtual systems', IFIP WG11.1&11.2, Amsterdam, Sept. 30 1999
- [FRI00a] Ann Frisinger, Louise Yngström 'An approach to use knowledge about user's security requirements in a risk analysis', accepted paper for IFIP World Computer Congress 2000 -- Information Security, Beijing, Aug. 2000.
- [FRI00b] Ann Frisinger, 'An assessment of how high level factors with influence on security in an organization differ from industry to industry', NORDSEC 2000, Oct. 12-13, 2000, Reykjavik, Iceland.
- [FOR98] Forrester Research Inc. report by Carl D Howewith, John C. Mc Carthy, Tom Buss, Ashley Davis, "Economics Of Security", February 1998.
- [HAM96] Gustaf Hamilton, 'Risk Management 2000', Studentlitteratur 1996, ISBN 91-44-00082-0.
- [HOW97] John D. Howard, 'An Analysis Of Security Incidents On The Internet', Ph.D. dissertation, Carnegie Mellon University, April 7, 1997, URL: <http://www.cert.org/research/JHThesis>.
- [JON99] Katarina Jonsson, 'Users' security requirements in the Networked Education process' master thesis, September 1999, DSV, Stockholm University

- [MAG99] Christer Magnusson, 'Hedging shareholder value in an IT-dependent business society', December 1999, Doctoral thesis, Stockholm University, ISBN 91-7265-011-7.
- [SIS99] SIS Svensk Standard SS627799-1: 'Ledningssystem för informationssäkerhet - Del 1: Riktlinjer för ledning av informationssäkerhet (Information Security management - Part 1: Code of practice for information security management)', 1999
- [TRU00] Truex, D., Baskerville, R., and Travis, J. 'Amethodical Systems Development: The Deferred Meaning of Systems Development Methods.', 2000, Accounting, Management and Information Technology, 10, 53-79.
- [VEN99] Hein Venter, Les Labuschagne, Jan Eloff, 'Real-time risk analysis on the Internet: a prototype', IFIP 1999, WG11.1 (Information Security Management) & WG11.2 (Small Systems Security), Amsterdam Sept 30 - Oct 1, 1999.
- [YNG96] Yngström Louise, 'A systemic-holistic approach to academic programmes in IT security', Oct. 1996, Stockholm University/Royal Institute of Technology, Report series No. 96-021, ISSN 1101-8526, ISRN SU-KTH/DSV/R--96/21--SE.

This Page Intentionally Left Blank

The Security Model to Combine the Corporate and Information Security

TEEMUPEKKA VIRTANEN

Helsinki University of Technology, Finland

Telecommunications Software and Multimedia Laboratory,

P.O Box 5400, FIN-02015, HUT, Finland

teemupekka.virtanen@hut.fi

Key words: security model, information security, corporate security

Abstract: There are two different areas of security that have been called the corporate security and the information security. Both of these areas have been organized in numerous ways. The lack of the common agreement of the security features and components has caused conflicts between security officers and IT people. Without a common structure it is also difficult to inspect and certify security. In this paper some of the existing approaches are presented and compared. This study is in a general level trying to find something one may call structure or organization of the security features. We then present a new model that combines the corporate security and the information security. This model is based on the assets and security measures. The measures are divided into six sections that may be easily organized in a typical corporation.

1. INTRODUCTION

There have been several ways to organize information security. A good model is a tool to plan, educate and organize security. To be useful the model has to fit its purpose. It has to be a good logical unit and help to manage the whole area. These models have to change according the development of the information technology, business logic and organization structure.

Security is a matter of culture, language and social environment [Yngström]. To be effective in improving the security the methods have to fit the local ways of thinking and organizing the work. The people also have to understand what they are doing and why. The lack of awareness of the users and managers is a problem that is very difficult to cover with technical equipments [James].

The security has to be a part of the business processes of the organization and people have a great role in these processes [Holbein]. The design of the processes is often a remarkable part of the assets of the organization. The documentation of these processes is often inadequate and lack of documentation is a threat for a organization.

There have been conflicts between security authorities and other managers in the organization. Security as a function covers so large area of the functions of the organization that situation is impossible without any restrictions. At the same time it is difficult to be sure that every aspects of the security have been taken into account if the definition or if the area of the security authorities has been reduced. Conflicts between security officers and information security officers are also common. The corporate security includes the information security and thus security officers tend to manage that area also. In the same time many areas of the information security belongs to the corporate security functions. So it is reasonable to assume that information security officers set demands on these functions.

There are several ways to organize corporate and information security. There might be a specific security department for the head of the security or security functions may be among the other administrative functions. Besides these alternatives the head of the information security may be also in the IT department.

When designing a secure system there have to be a model of security [Eckert]. Many of the models considered information security as computer security or information system security [Rannenber], [Eloff]. In the e-commerce applications one has to take also other than technical aspects into account [Labuschagne], [De Win].

In this paper several security/information security models are presented. All the models are designed for specific purpose and that those purposes naturally have their effects in the model. Every organization has to select the model suitable for its needs and structure. The purpose of this paper is however to present a new model that combines the best parts of these models and specifically combine the corporate and information security into one solid entity.

2. SOME CURRENT MODELS

2.1 The Finnish governmental policy

The actual model described here is from the Finnish governmental order about information security in the administration. This order was first time given in 1992 [Fin92] and revised in 1998 [Fin98].

The protection measures are divided into eight sections: **Administrative security** includes policies, principles, organization, duties, education and monitoring. **Personnel security** tries to minimize personnel risks with background checking, duties sharing, authorizations, personnel protection, education and monitoring. **Data security** includes availability, integrity, confidentiality and secure handling of the data. **Physical security** (environmental security) means protecting equipments, data and computer rooms from physical threats and damages. **Hardware security** includes all the security measures implemented by computer hardware. **Software security** includes all the security measures implemented by computer software. **Communications security** includes all the security measures that ensure availability, integrity and confidentiality of the information in the communication networks. **Operations security** is a security of daily duties (Figure 1).

2.2 ISO-model (BS7799)

The British Standard is based on PD 0003, a Code of practice for information security management, developed by the Department of Trade and Industry with the assistance of leading UK companies and organizations. The guidance in this code of practice is intended to be as comprehensive as possible. Not all controls described will be relevant to every situation. [ISO]

Security has to be embedded in the every day's business processes, with responsibilities for all people involved. The Code of Practice for Information security Management gives one practical guidelines in designing, implementing and assessing information security measures, in a technical, physical and organizational sense.

The code of practice is divided into 10 sections as follows: **Security policy, security organization, classification and control, personnel security, environmental security, network management, system access control, system development and maintenance, business continuity planning and compliance**

Besides these sections there are 10 key controls that are either essential requirements or are considered to be fundamental building blocks for information security.

2.3 CobiT-model

Information Systems Audit and Control Foundation has made CobiT to help bridge the gaps between business risks, control needs and technical issues. It provides good practices across a domain and process framework and presents activities in manageable and logical structure. [COBIT]

To satisfy business objectives, information need to conform to certain criterias, which CobiT refers to as business requirements for information. There are quality requirements (quality, cost and delivery), fiduciary requirements (effectiveness & efficiency of operation, reliability of information and compliance with laws and regulations) and security requirements (confidentiality, integrity and availability).

The CobiT framework consists of high-level Control Objectives and an overall structure for their classification. There are three levels of IT efforts. Starting at the bottom, there are activities and tasks needed to achieve the measurable result. Processes are then defined one layer up as a series of joined activities or tasks with natural control breaks. At the highest level, processes are naturally grouped together into domains. Their natural grouping is often confirmed as responsibility domains in an organizational structure and is in line with the management cycle or life-cycle applicable to IT processes.

The IT resources identified in CobiT are data, application systems, technology, facilities and people.

The definitions of the four domains are:

Planning and Organization : This domain covers strategy and tactics, and concerns the identification of the way IT can best contribute to the achievements of the business objectives.

Acquisition and Implementation: To realized the IT strategy, IT solutions need to be identified, developed or acquired as well as implemented and integrated into business process.

Delivery and Support: This domain includes the actual processing of data by application systems, often classified under application controls.

Monitoring: This domain addresses management's oversight of the organization's control process and independent assurance provided by internal and external audit (Figure 1) .

2.4 The Finnish corporate security

The Confederation of Finnish Industry and Employers has made several publications for the Finnish companies. A corporate security is one of the areas of those publications. In these publications there are a model in which corporate security is divided into several areas. This same model is also used by the Lifelong Learning Institute Dipoli (Helsinki University of Technology, HUT), in its education program for security managers. [Pesonen]

The corporate security is divided into several different areas. This dividing is based on the legislation of Finland that sets some security related duties to companies. Typically there must be a person in the company who is responsible to arrange these duties. Then there has to be a learning material and a course for these persons and that creates a new area into the security model (Figure 1).

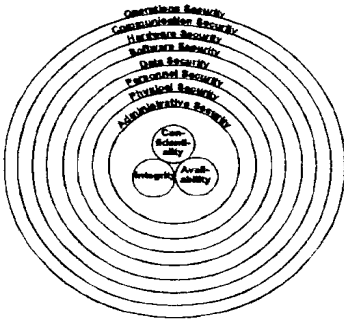
Administrative security includes the security related legislation, security standards, relationship with officials, recruiting (security related matters) and PR (security related matters). **Physical security** includes buildings and the security monitoring in these buildings. Most of the crime preventing and monitoring activities in the building are part of physical security. **Fire prevention** and shelters includes fire prevention in buildings, fire extinguishers, fire detectors and shelter rooms for people **Safety** includes safety in the work and the health services. **Personnel security** includes the protection of own personnel and customers. Especially the key persons must be protected also during travel and in abroad. **Environmental protection** includes pollution prevention and the handling and storing of the dangerous or hazard chemicals. **Transportation security** includes the crime prevention during transportation and prevention of the road accidents. **Security abroad** includes the local legislation, connections to the local authorities and all the local actions. **Emergency Supply** includes sufficient facilities must be built up and maintained to ensure the production and output of goods and services in a wide spectrum of industries, in times of unusual conditions. [Supply Act]. **Information security** includes handling of the printed material, archiving, computer security, communication security and privacy protection. **Risk management** includes methods to manage risks, security analyses and insurances

There is not any connection to the goals of the organization and thus the corporate security according this model may fulfill mainly requirements that come outside the organization.

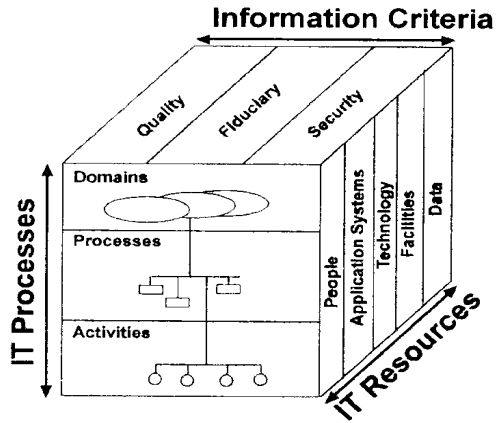
3. ANALYZING THE CURRENT MODELS

3.1 Analyses of the models

All these security models described in the previous section have a specific purpose they are made for. Therefore one can't say they are wrong or have problems. However, it is difficult to create a solid model based on these models.



The Finnish governmental model



The CobiT-model



The Finnish corporate security

Figure 1. The current security models

The Finnish governmental policy for information security is old-fashioned. It fits better in the time of mainframes than modern client-server architecture where one can't make difference between hardware, software and network.

The BSI-standard is a list of useful measures to improve information security and it makes very easy to qualify the level of the organization. Dividing these measures to the sections is not made systematically and therefore there is not any logical model of information security.

The Cobit-model has a strong live-cycle background. Every information system has a designing-, implementation- and daily work -phase. Inside these domains there are the processes where the actual security measures are. Each process has a list of resources, requirements and information criterias it fulfills. There are several points of view in these criteria and they are not formed a simple model. One can make this approach better by simplifying it more.

The Finnish corporate security –model is a model for a corporate security as a whole. The information security is only a part of it. I have taken it into this paper because I compare it with the model of pure information security and try to combine these both. The problem of this model is that there is not a proper connection between business needs and security. This model is made to fulfill the requirements coming from outside the organization. There is no coordination of these requirements and therefore there are many conflicts.

3.2 Corporate security and information security

There may easily be conflicts between corporate security and the information security if the responsibilities are not clear. One can ask if the information security is a part of the corporate security or vice versa. There have also been issues about whether the manager of the information security should be in the computer department or security department.

If we look at both the Finnish models, one for the corporate security and other for information security, we may notice that there are many common elements. In the corporate security there is one section for information security and in the information security there are several sections that exist also in the corporate security.

This conflict is based on the fact that actually the information security as a term is a combination of goal and measures. Basically the same conflict exists in the protection of environment. Graphically we can try to modify the original pie chart taking the information security out from the pie to the new layer in the top of the pie.

4. THE NEW MODEL

The basic principle in the new model is that there are assets, threats and measures. The assets must be recognized and classified. The threats must be found out and analyzed. The measures must find out to prevent the threats or minimize the damages they caused.

4.1 The assets

There are four different types of assets: information, people, material and goodwill. These are not totally independent each other because people might actually be a storage of information and a pair of hands like a machine. If people are considered this way however, it will cause damage to the goodwill.

The value of the asset may come from several sources. It may be given outside the organization, like in the legislation, or the item may be crucial for the working process. Some items may have pure monetary value.

All the assets may be valuable by several ways. The confidentiality, availability and integrity are not only properties of information but the general classifying criteria. In a real life however, the availability is the most important property of all the other assets than information. In this context monetary value is same as the availability (to sell).

4.2 The measures

The measures are divided into six sections. They are

Security management is a link between business objects and security. In this process the security level needed for business is decided and also what is the risk an organization may take. The goals and the visions of the organization have an immediate effect to the security level. This is also the place to organize security functions in the organization.

Asset management is a process where the assets are found and classified.

Physical security (environmental security) prevents any physical threats that come from outside. There might be several security areas in the organization and everything that leaves one area and go into another must go through barrier of physical security. Normally physical security cannot help against a threat that already is in the area, but there are some exceptions: fire prevention systems work inside the perimeter and also the guard.

Personal security differs from the earlier models because it is not protecting people. Personal security protects all the assets using methods that affect people. In a recruiting situation one has to prevent the unsuitable

people to be hired. The employees have to be capable and willing to work secure manner.

Computer security includes all the security functions of normal computers and communication. It might be implemented by software, hardware or separate devices. If the computer is built into special system so that it is invisible to the user or it is impossible to program, it is handled like a system without a computer. For example a fire detecting system may have a computer inside but because the computer can't do anything else control the system, it is not part of the computer security but physical security.

Work security connects security to the daily work. It is like a quality in the work avoiding accidents that may damage the assets. The routines of the work are planned so that there are no unnecessary losses.

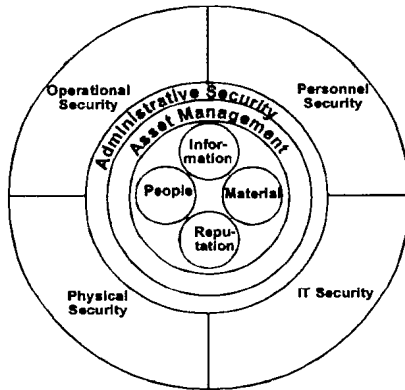


Figure 2. The new model

In a typical situation there is a room as a single security area. Security management has told what is the acceptable level of risk in this company. Asset management has told which items in the room are valuable. Personal security selects the people who are allowed to enter and educates them to work properly. Physical security prevents others to enter. Work security guarantees that the normal work of the allowed people do not cause any damage to the assets. Computer security takes care of the electrical universe. There might be connections to and from the room and computer security watches these. In the room there might be another room with its own security level and authorized people.

5. THE ADVANTAGES OF THE NEW MODEL

In this model there is clear solution between security and information security. There is no information security. There are only several protecting

methods that are applied to secure assets. Information is one of the assets and is protected in a same way as the other assets are.

This model helps to avoid duplicate protecting systems. The existing system may protect all the assets and there don't have to be separate systems for different assets.

The sections are as independent each other as possible. This avoids conflicts between different parts of the organization.

The model is logical and easy to understand. That makes easy to describe the elements of the security and teach them to others.

The measures are easy to organize. Security management is a part of typical business management. Most of the decisions required are the same that are made when decide the business goals and methods. Personal security is a normal work of personal department like the computer security belongs to the computer department.

The physical security is a concrete section of security. In many cases there already is a security department managing these duties. However there might also be a department taking care of buildings and some of the duties in the physical security might have been there.

The work security is based on the normal hierarchy. A supervisor has to be sure that subordinates work in a secure way. These standards have to be essential part of any goal settings or orders. The work has also to be organized in a secure way. The worker has to have time and resources to finish the job without breaking the rules. The working procedures have to contain checking points and inspections to guarantee the quality of the work.

6. CONCLUSION

In this paper a new model is presented to organize security and information security. In this model the both areas are combined into a corporate security.

The assets are people, material, information and reputation. The value of the asset may be based on the availability, confidentiality or integrity of the asset. The classification principles may come from outside (legislation, reputation) or inside the organization (production requirements).

The reputation might be considered as a special case that is based on the compliance of the rules and regulations and might also achieve by adding the assets on outside the organization, for example instead protecting the own people only protect also the people in the neighborhood.

The protecting measurements are divided into four areas. This dividing is based on the normal organization of the corporation. This makes security

functions as a part of the normal work of the people that normally carry out these tasks.

The model has been tested by using it some years in two nationwide organizations. It has simplified the security organization and made it easier to write down security policy and requirements. One of these organizations has a security department that has divided into several sections according the functions. These sections coordinate the security functions in the other departments (personnel, computer, real property) and line organization. The other corporation has only a security manager that coordinates the security work in the corporation. This manager sets the standards of the corporation and consults the departments in the corporate headquarter and management in the divisions.

REFERENCES

- [COBIT] CobiT Framework 2th edition, <http://www.cobit.com>, 1998
- [De Win] De Win, B., Van den Bergh, J., Matthijs, F., De Becker, B., Joosen, W.: "A security architecture for electronic commerce applications", Information Security for Global Information Infrastructures (IFIP SEC 2000), pp. 491 – 500, Kluwer Academic Publisher, The Netherlands 2000, ISBN 0 7923 7914 4
- [Eckert] Eckert C.: "On Security Models", Information Systems Security (IFIP SEC '96), pp. 485 – 486, Chapman & Hall, UK 1996, ISBN 0 412 78120 4
- [Eloff] Eloff, M.M., von Solms, S.H.: "Information Security: Process Evaluation and Product Evaluation", Information Security for Global Information Infrastructures (IFIP SEC 2000), 11 – 17, Kluwer Academic Publisher, The Netherlands 2000, ISBN 0 7923 7914 4
- [Fin93] Valtioneuvoston periaatepäätös tietoturvallisuudesta (In Finnish), Ministry of Finance (of Finland), Finland 1993
- [Fin98] Valtioneuvoston periaatepäätös tietoturvallisuudesta (In Finnish), Ministry of Finance (of Finland), Finland 1998
- [Holbein] Holbein, R. et.al: "The use of Business Process Models for Security Design in Organisations", Information Systems Security (IFIP SEC '96), pp 13 – 22., Chapman & Hall, UK 1996, ISBN 0 412 78120 4
- [ISO] British Standard, Code of Practise for Information Security Management, BS7799:1995, British Standard, ISBN 0 580 23642 0, 1995
- [James] James, H. et. al: "A human approach to security management in HealthCare" Information Systems Security (IFIP SEC '96), pp 365 – 376, Chapman & Hall, UK 1996, ISBN 0 412 78120 4
- [Labuschagne] Labuschagne, L: "A framework for electronic commerce security" Information Security for Global Information Infrastructures (IFIP SEC 2000), pp. 441 – 454, Kluwer Academic Publisher, The Netherlands 2000, ISBN 0 7923 7914 4
- [Personen] Pesonen Markku: Yrityksen turvallisuusjärjestelyt (Security Arrangements of a Company), ISBN 952 90 4705 3, Yliopistopaino, Finland, 1993
- [Rannenbergl] Rannenbergl Kai: "IT Security Certification and Criteria", Information Security for Global Information Infrastructures (IFIP SEC 2000), pp 1 – 10, Kluwer Academic Publisher, The Netherlands 2000, ISBN 0 7923 7914 4

[Supply Act] Security of Supply Act, The Ministry of Defence, 1390/92, 1992

[Yngström] Yngström L.: "IT Security and Privacy Education", Information Systems Security (IFIP SEC '96), pp. 351-364, Chapman & Hall, UK 1996, ISBN 0 412 78120 4

Design Criteria to Classified Information Systems Numerically

TEEMUPEKKA VIRTANEN

Helsinki University of Technology, Finland

Telecommunications Software and Multimedia Laboratory

P.O. Box 5400, Fin-02015 HUT, Finland

teemupekka.virtanen@hut.fi

Key words: security, classification, evaluation

Abstract: Constant changes in the structure of the organization and the working processes have forced security staff to reclassify and re-evaluate information and information systems too often. In this paper we present one solution to make it possible to use the previous data as much as possible and recalculate the evaluation results automatically.

The solution is based on piercing the processes into parts of the block diagram and then analyzing the classification of the each block. This procedure is continued from top to down until there is no remarkable processes left. After the top-down phase has been reached its end a second phase is started from bottom to top. In this phase the reliability of each block is analyzed and the results of one level is combined. This result is then passed to the upper level and this procedure may continue until the top is reached.

In every level it is possible to have iterative loops if the requirements are not met. It is usually easier to add parallel processes for assurance than improve the reliability of the single component.

1. INTRODUCTION

The classification of computer systems have been described in a qualitative way [Tryfonas] that promotes availability because it has attracted the less attention than the other parts of CIA-model triplet (confidentiality and integrity).

There have also been papers that describe availability using the techniques from availability engineering [Lyu]. Many of these assume information systems as a single system or focuses on the reliability of the software [Herrmann], [Leveson], [Kapur]. Information system is, however, an information handling process that may require several attendants and manipulating systems [Kiountouzis]. The information may have a working flow that directs some parts of the information to one department and the rest to another [Smith].

Availability in the information security differs from the reliability in engineering in the sense that in the reliability engineering everything is statistical. In the availability systems are often targets for the attacks and therefore statistical approach is often useless. Information systems may be considered as open systems [Leveson], as they communicate with their environment in a flexible way. This kind of system is often unstable instead of the stable systems where the communication with the world is reduced and formal.

There are however several advantages to use numerical methods. There must be some policies to establish the connection between numbers and the real life but using numbers it is easier to use computers and automatic procedures to handle the information. There are however few methods to describe the other areas than availability with numerical methods [Jønsang].

The delegation and outsourcing have been became popular ways to organize the work of the organization. This requires a proper way to pass the classification of the information and procedures with the delegation. There are often in the each level possibility to organize the work of its own. For the verification there has to be a method to pass the result of the security analyze to the superior level and summarize them.

2. NUMERICAL CLASSIFICATION METHOD

2.1 Top Down system classification

A classification is a part of business management. In a top level has to be known which processes are the important ones for the organization. In a highest level this may set the standards for the division of the organization saying what is the meaning of the division to the company and what are the remarkable products of the corporate.

In the next level the head of the division has to classify the business processes that are essentials to achieve the goals of the division. This process

may continue until the bottom level is reached or there is another termination rule.

In the classification all the aspects of information has to be notified. In approach like this the availability is a natural aspect but in the same time a higher level has to determine the confidentiality and integrity class of the information it gives to the lower level to be processed.

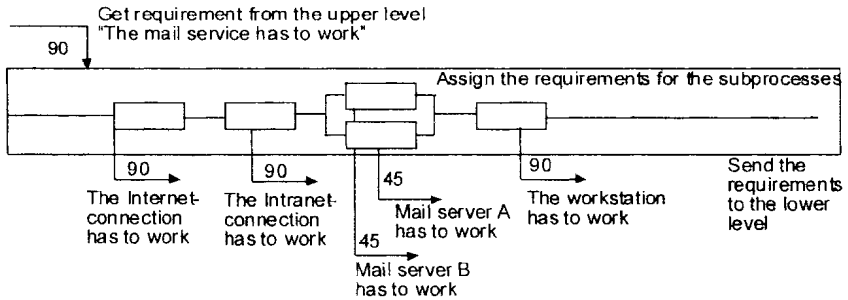


Figure 1. The classification process

The subordinates (lower level) get these classifications as an order. If they like to receive the information they need, there must be proper security methods to handle the information. In the same way a client (upper level) sets the availability standards and requirements for the quality of the information.

2.2 Bottom up system evaluation

The requirements are set from top to bottom. The results of the evaluation processes go from bottom to top. In every level the results of its subprocesses are combined and the result is then delivered to the superior level.

In the lowest level the evaluation may check the reliability of certain components or devices. In the upper levels the result of the evaluation is typically information that is summarized from results of the subprocesses and the subresults. This means that the hard work is done in the bottom level. All the evaluation information in the upper level is already numerical and easy to recalculate if needed.

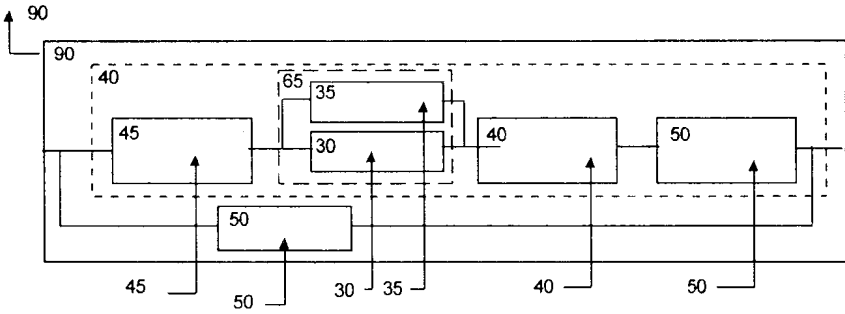


Figure 2. availability evaluation

In every level the results have to be compared with the classification. If the results don't meet the classification, there has to be an attempt to correct situation. This may happen either requesting the lower level to make it's result better or adding redundancy by reorganizing the work flow and adding some parallel subprocesses..

3. AVAILABILITY

In this paper availability is the probability that a process will produce its result in the designed time. The scale is 0-100 where 0 means that process will never succeed in time and 100 means totally always. We understand that there are other possible definitions. Especially the delay is also important. We have to assume that a process with high probability have also very short delay those cases it misses the designed time.

The required classes and their limits are defined in the information security policy of the company. One may for example define that there are four classes: top important (100-80), important (80-60), useful (60-40) and no requirements (40-0). In this definition a process that is classified as a top important must succeed to produce it's output in defined time four times out of five. In the policy there might also be defined a lowest level for the classification and evaluation. In the previous example the natural lower limit is 40 because systems below this level are considered useless or at least there are no requirements for systems below that.

In the classification procedure every piece of process gets an availability requirement from its superior process. That is the goal this process has to fulfill. This procedure takes automatically account of the business processes because the requirements come from the top. That means the more important a process is for business the higher requirements should be set for it.

In the next step the process has to be analyzed and divided to the subprocesses. These subprocesses may follow each other when they are said to be in series or they may be alternatives when they are said to be parallel. This analysis produces a block diagram where are the subprocesses and their connections.

Each subprocesses must be classified. Each subprocess in a series must have the same classification and that must be the same as is required for a whole block. If there are subprocesses or blocks in parallel, they must have classification where the original requirement is divided with a number of parallel blocks. In some cases this automatic procedure is not practical. If one of the parallel processes is the most profitable choice and the other are the more expensive backups it is reasonable to set the higher requirements for this alternative than the others.

When each subprocess has the classification, it is passed to the subprocess that means the classification steps one level downwards. This procedure is repeated until the classification of the subprocesses reach the termination level that means the lower processes are not significant.

In the second phase the evaluation is performed from bottom to top. In each level the availability of it's subprocesses are measured using the same scale as in the requirements. In the lowest level this may be the reliability of the processing equipment and in the upper level it is the results of the lower level.

The results are put in the block diagram that describes the process and the evaluation of the process is summarized using the same formulas as in the requirement phase. The result is the evaluated availability of the process and is compared with the requirement. If the result does not meet the requirement, corrective measures have to be started.

There are two ways of improving the availability. One can improve the reliability of the subprocess or add parallel subprocesses. The improving means that a new higher requirement is sent to the lower processes and those have to make their arrangements to meet these new requirements. Adding new parallel subprocesses means that the requirements for each parallel subprocesses will be lower.

It is easy to see that it is overall much more easier to add some additional procedures for the situation the main process is not available. The parallel availability is much more efficient that trying make single process more reliable.

4. INTEGRITY

For the integrity the basic procedure is the same as for the availability. Each level gets the requirements from the upper level, analyzes the graph of its own functionality and makes a block diagram of its subprocesses. An integrity requirement is then assigned to each of these subprocesses and passed to the lower level.

After the termination point has been reached the evaluation starts and produces an integrity level that is passed to the upper level. If the criteria do not meet, the corrective measures will be started.

In this paper the integrity means the quality of the information. The integrity is high if the information was correct in the beginning and the correctness of the information was ensured during the processing so that it is correct in the end of the processing. The integrity decreases always when the original information is changed. If there is proper assurance for the changes to be correct the integrity doesn't decrease.

The classes and their definition are defined in the information security policy of the organization. They might be for example high integrity (100-70), normal integrity (70-40) and no requirements (40-0). There are no clear correspondence between this number and the real life like in the availability case.

In the block diagram subprocesses in series means steps that handles information straightforward and in the parallel subprocesses the information are cross-checked. Thus the serial coupling decreases the integrity and the parallel coupling increases the integrity. The processes that are read-only don't change the integrity of the data.

5. CONFIDENTIALITY

For the confidentiality the basic procedure is the same as for the availability and integrity. Each level gets the requirements from the upper level, analyzes the graph of its own functionality and makes a block diagram of its subprocesses. A confidentiality requirement is then assigned to each of these subprocesses and passed to the lower level.

After the termination point has been reached the evaluation starts and produces a confidentiality level that is passed to the upper level. If the criteria do not meet, the corrective measures will be started.

In this paper the confidentiality means the secrecy of the information. The requirements for the systems are based on the confidentiality of the information the system processes. The secrecy is based on the fact how much damage it is caused if the information is came public to unauthorized

people. This amount of damage may be based on the loss of money or time to correct the situation with new information.

The classes and their definition are defined in the information security policy of the organization. They might be for example top secret (100-80), secret (80-60), confidential (60-30) and unclassified (30-0).

When the block diagram is made one has to concentrate the information that is required to pass to the subprocess as an input. If the subprocess requires the information it has to be authorized to receive it. This authorization is achieved by setting the confidentiality requirements for the subprocess.

The confidentiality of the information may be decreased by dividing it into pieces. The part of the information is often less confidential as the whole information. There are two ways of decreasing the amount of the information in one single point. One may decrease the amount of the information, for example records in the database, or quality of the information, for example fields in the database.

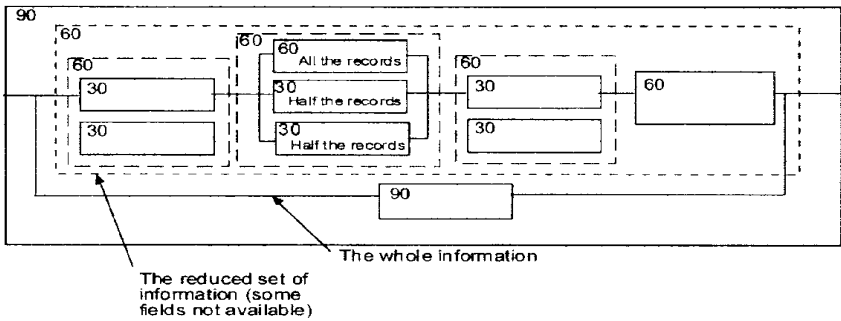


Figure 3. Confidentiality classification

In the block diagram subprocesses following each other means that they use the same information and thus must have a same confidentiality classification. If the information is the same that comes from the upper level, the classification must be the same as the whole process classification. If the information is divided into smaller units then the classification of the subprocesses must correspond the amount of information they receive.

One has to notice that the classification process requires intelligence. This means that somebody has to make the block diagram and set the classification. After this work has been done it is possible to use the information automatically by computers to recalculate the results. It is also

possible to redefine one piece of diagram and recalculate the whole result without any other changes.

6. THE ADVANTAGES OF THE NEW METHOD

The new model is designed for the modern business organization. An upper part of the organization may assign goals to its subordinates and delegate the decisions and resources needed to fulfill the goals to the subordinates. This model clarifies the goals of the security function so that it may also be delegated to the subordinate. The lower level may organize its own processes in a way it can fulfill the requirements.

In this model the classification and evaluation of the security is hidden in the black boxes of the subprocesses. The upper level doesn't have to know how the results are achieved. It has to trust the lower level.

When the classification and the requirements are clear it is possible to outsource the parts of the processes. The information and service level between the organizations are clarified and classified.

The new model makes it possible to program all the processes and their relationship into a database and automatically calculate the security level of the organization. When the block diagrams are in the database, all the modifications are local and the whole database can be easily recalculated. The changes are local and they affect only local diagram. The changes of the numerical values are summarized automatic.

The department level it is easy to manage the situation. The requirements are defined in the upper level and the results of the evaluation are collected from the lower level. Both requirements and results are in the common form and there are defined functions to produce them.

7. CONCLUSION

The new organization structures causes problems for the classification and evaluation processes because all the changes requires reclassification and re-evaluation. The new method described in this article makes it possible to reorganize the processes with local changes and automatic recalculation.

For the calculation the classification and the evaluation have to be numerized. We present one possible methods for this. This classification and evaluation are carried out in the basically same way. The exact numeral values and their explanations have to be described in the security policy of the corporation.

The delegation and outsourcing of the processes are easy when the interface is clear. The subordinate may arrange its work independently as long as it meets the requirements.

The results of the evaluation may be improved using two methods. Either one has to improve the reliability of the single component (device or subprocess) or one has to decrease the importance of the component by adding a parallel process for assurance. One level may add these parallel processes independently from other parts of the organization.

8. REFERENCES

- [Herrmann] Herrmann D.S: "Software Safety and Reliability", IEEE Computer Society Press, USA 1999, ISBN 0-7695-0299-7
- [Jønsang] Jønsang, A., Knapkog, S.J.: "A Metric for trusted systems", Proceedings of the IFIP SEC 1998, Chapman & Hall, UK 1998
- [Kapur] Kapur P.K., Garg R.B, Kumar S: "Contributions to Hardware and Software Reliability", World Scientific Publishing Co. Pte. Ltd, Singapore 1999, ISBN 981-02-3751-0
- [Kiountouzis] Kiountouzis, E.A., Kokolakis, S.A.: "An analyst's view of IS security", Information Systems Security (IFIP SEC 1996), Chapman & Hall, UK 1996, ISBN 0 412 78120 4 [Tryfonas] Tryfonas T, Gritzalis D, Kokolakis S: "A Qualitative Approach to Information Availability", Information Security for Global Information Infrastructures (IFIP SEC 2000), Kluwer Academic Publisher, the Netherlands 2000, ISBN 0 7923 7914 4
- [Leveson] Leveson N.G: "Safeware – System Safety and Computers", Addison-Wesley Publishing Company, USA 1995, ISBN 0-201-11972-2
- [Lyu] Lyu M: "Handbook of software reliability engineering", McGraw-Hill, 1996
- [Smith] Smith, E., Eloff, J.H.P: "Modelling risks in health-care institution", Proceedings of the IFIP SEC 1998, Chapman & Hall, UK 1998

This Page Intentionally Left Blank

Deception: A Tool and Curse for Security Management

M Warren¹ and W Hutchinson²

¹Dept of Computing & Mathematics, Deakin University, Geelong, Victoria, Australia 3216.

²School of Management Information System, Edith Cowan University Churchlands Western Australia, Australia, 6018.

Key words: Deception, Information Security, Security Management.

Abstract: With the proliferation of electronic information systems over the last two decades, the integrity of the stored data and its uses have become an essential component of effective organisational functioning. This digitised format, used in input, output, processing, storage, and communication, has given those wishing to deceive new opportunities. This paper examines the nature of deception, and its potential as a new security risk in the information age.

1. INTRODUCTION

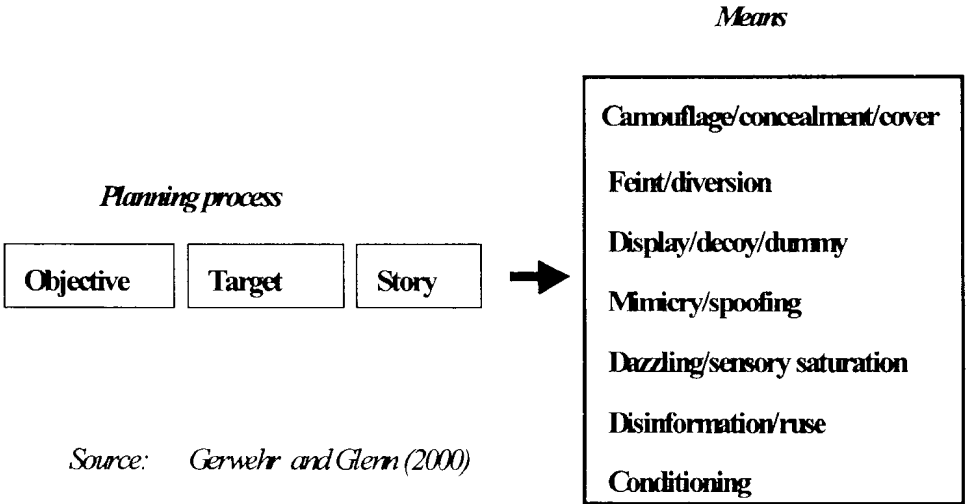
One major advantage of the digital media is the ability to easily manipulate the bits that constitute its messages. It is also one of its major disadvantages. For instance, Roberts and Webber (1999) trace the history of photographic manipulation and clearly show the ease with which images can be changed to give a totally different perspective. In the digital realm, this is sold as one of the major advantages of computerised imagery. Photographic images, which were always slanted versions of reality, cannot even be taken to be that in today's digitised world. Any component of the image can be changed to reflect whatever is required. Barry (1997) demonstrates the power of visual imagery and how subtle changes can disproportionately change the meaning of an image. Brugioni (1999) illustrates that there is no shortage of

government and private organisations as well as individuals only to willing to apply photo-fakery. This is just as appropriate with simpler, conventional text messages. It can be imagined the damage caused to an organisation if a Web based employment advertisement phrase was changed from “Applications from all ethnic groups welcome”, to “Applications from all ethnic groups not welcome”. This, not so subtle change, could easily cause the organisation involved an enormous amount of embarrassment with its inherent use resources to rectify the situation. Contemporary organisations with their reliance on information technology are vulnerable to deception. Of course, this technology also provides an opportunity. Each person or group can become a deceiver as well as being a victim of deception. Any management regime needs to be fully aware of the potential for deception, and its potential impacts on organisation decision making and operations. Manipulating data to produce desired outcomes has been routinely practiced since the dawn of history. Individuals and organisations choose data which suits the image they want to be portrayed, soldiers camouflage weapons to avoid detection, or disperse false information to conceal intentions. In simple terms, the function of security is both to protect assets and avoid deception from manipulated data.

2. PRINCIPLES OF DECEPTION

In this paper, deception is defined as *the deliberate alteration of data or a situation's context to promote a desired outcome*. Therefore, it does not include self-delusion, or a person's natural tendency to use mental model to interpret things in an individual way. The definition places emphasis on a second party being involved, where that person or organisation is consciously trying to create deception.

To understand the fundamental of deception, it is necessary to define data, information, and knowledge. Boisot's (1998) model defines data as the attribute of a 'thing' such as, its colour, shape, or its value. Knowledge is an attribute of an 'agent' (usually this means a human, although it can be argued that intelligent machines can have knowledge). Knowledge is a product of experiences, education, age, gender, culture, and many of the other factors that make up individuals. Thus, humans derive information by using their knowledge to select appropriate data to provide them with information. Hence to deceive, it is necessary to alter data by addition, deletion, or modification and/or alter the context in which the data is interpreted.



Source: Gerwehr and Glenn (2000)

Figure 1. Types of Deception

Bowyer (1982) classifies deception into two main types that of Level 1: Hiding the real and Level 2: Showing the false. It should be pointed out that ‘showing the false’ also involves ‘hiding the real’. Figure 1 details the types of deception. Whilst this paper is too short to go into each method of creating an illusion by ‘feeding’ data to an unsuspecting person, the variety of techniques to do can be left to the imagination. Also, there is the potential to manipulate the context by which data is interpreted. Deception is an option for both attacker and defender alike. This paper will consider both but further discussion of Web based deception can be found in Hutchinson and Warren (2000a, 2000b). It can be seen from figure 1 that for effective deception an objective, a target, and a story are required. A method of achieving the objective needs to be decided. As mentioned before, these can be used by the security function or against it.

3. USING DECEPTION TO AID THE SECURITY FUNCTION

Many technical deception systems are used by the security function to deceive individuals in order to obtain information about their actions. These on-line tools are used to deceive hackers into thinking they are attacking an actual system, instead all their activities are being recorded. Some commonly used approaches are:

3.1 Honeypots

A *honeypot* is a ‘pretend’ server with the aim of tracking *black-hats* (an unauthorized person trying to get access to a system (Spitzner, 2000a) in the act of probing and compromising a system. The aim is to deceive the black-hat into thinking they are attacking an actual real life server (software examples include systems by Cohen (2000), and Network Associates (2000)). The aim of the honeypot is to monitor the black hats by a number of means (Spitzner, 2000a), they are:

- Tracking the honeypot firewall logs
- Analysis of honeyPot system logs to determine what the kernel and user processes are doing.
- Using a *sniffer* on the firewall that ‘sniffs’ any traffic going to or from the honeypot. The advantage of a sniffer is that it picks up all keystrokes and screen captures.
- Using a *tripwire* on the honeypot. A tripwire tells the system administrator what binaries have been altered on a compromised system (such as a new account added to: /etc/passwd, or a trojaned binary).

The aim of the honeypot is to attract the black-hats, monitor them, let them gain root access to the system, and then eventually log them off the system, all without any suspicion being aroused. Once black-hats gain root access, they are monitored for several days in order for the system administrator to learn what they were doing. The biggest problem is how to limit the black-hats offensive actions (Spitzner, 2000b).

This is done by using the honeypot firewall, and implementing a rule base schema that allows access from the Internet to a honeypot’s firewall, but limits outbound network traffic. It is important that the black-hat is allowed enough outbound traffic so as not to arouse suspicion. The results of these honeypot assessments are made public (<http://project.honeynet.org/>) so that network administrators can access the information and ensure that they are protected against common hacker attacks and techniques. The following figure illustrates the output recorded by a hacker trying to attack a Honeypot. (*ibid*).

```

!"" #!""# ' 9600,9600VT9111VT9111
Red Hat Linux release 6.0 (Shedwig)
Kernel 2.2.5-15 on an i586
apollo /]# TERM=vt9111
telnet ns2.cpcc.cc.nc.us
ns2.cpcc.cc.nc.us: Unknown host
@apollo /}#telnet 1 152.43.29.52
Trying 152.43.29.52...
Connected to 152.43.29.52.
Escape character is '^]'.
!!!!!!Connection closed by foreign host.
te8ot@apollo /]# TERM=vt7877
[root@apollo /]# telnet sparky.w
itoot@apollo /]# exit
exit

```

Figure 2: The output from a honeypot of a Hacker's attack

The information obtained from the honeypot audits are forwarded to CERT (<http://www.cert.org/>) for their assessment and also the system administrators of the systems involved in the attack.

3.2 Honeynets

The work by Spitzner developed into expanding the honeypots into *honeynets*. Spitzner (2000c) identified that the honeypots needed to be expanded for the following reasons:

- to be able to determine attacks upon switches, routers and different operating systems of a network
- generate information from several sources (for example, honeypots) in order to provide information in greater detail.
- detect new attack patterns such as vulnerability scanning and how black-hats progress from one system to another.

The result was grouping a number of honeypots together to form a honeynet, so a black-hat would feel that they were gaining access to a much large networked system. When in reality more of their actions and attack strategies will be recorded.

4. SPOOFING ATTACKS

Attackers also use deception. For instance, in a web *spoofing attack*, the attacker creates an on-line environment within which a victim will be deceived and disclose information such as passwords. The secret of web-spoofing attacks is to create an environment in which the victim is misled into thinking they are actually at the correct web-site and undertaking actual transactions. To start an attack, the attacker must somehow lure the victim into the attacker's false on-line web site.

There are several ways to do this. An attacker could put a link from a popular web page to a false web page. If the victim is using web-enabled email, the attacker could email the victim a pointer to a false web site, or even the contents of a page in a false web site. Also, the attacker could trick a web search engine into indexing part of a false web site. The key to this attack is for the attacker's Web server to sit between the victim and the rest of the web (Felten et al, 1997).

The attacker's first trick is to rewrite all of the URL (Uniform Resource Locators) on a web page so that they point to the attacker's server rather than to some real server. "Assuming the attacker's server is on the machine www.attacker.org, the attacker rewrites a URL by adding <http://www.attacker.org> to the front of the URL. For example, <http://www.home.netscape.com> becomes <http://www.attacker.org/http://home.netscape.com>" (from Felten et al, 1997, p.3).

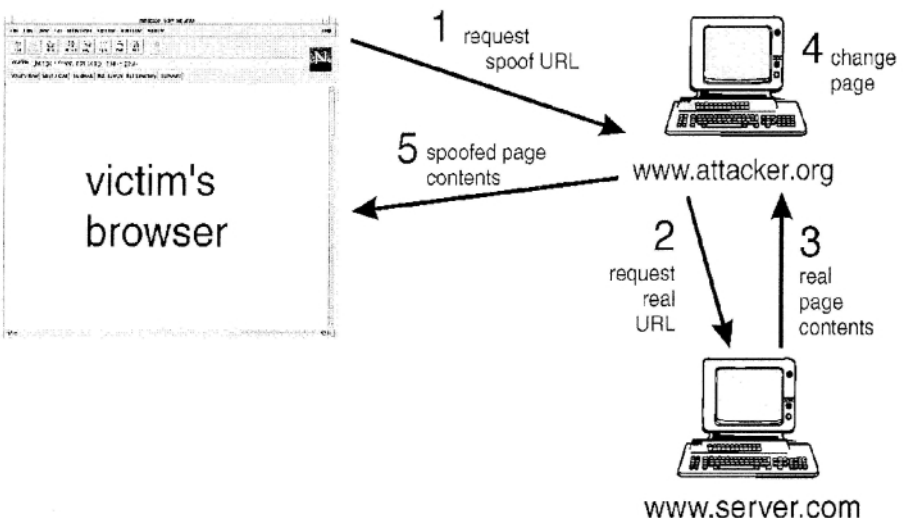


Figure 3: Examples of Web Spoofing Attack (from Felten et al, 1997)

Figure 3 shows an example Web transaction during a Web spoofing attack. The victim requests a Web page. The following steps occur: (1) the victim's browser requests the page from the attacker's server; (2) the attacker's server requests the page from the real server; (3) the real server provides the page to the attacker's server; (4) the attacker's server rewrites the page; (5) the attacker's server provides the rewritten version to the victim.

Figure 4 illustrates a real life example of web-spoofing. This service is offered by Anonymizer (<http://www.anonymizer.com/>) and offers anonymous viewing of web pages by the use of web spoofing, in this example the Web page of the Australian Broadcasting Corporation is viewed by the Anonymizer server, the implication of this is that any IP (Internet Protocol) logging tools would track <http://www.anonymizer.com/> but not <http://www.abc.net.au> which would protect the privacy of a user.

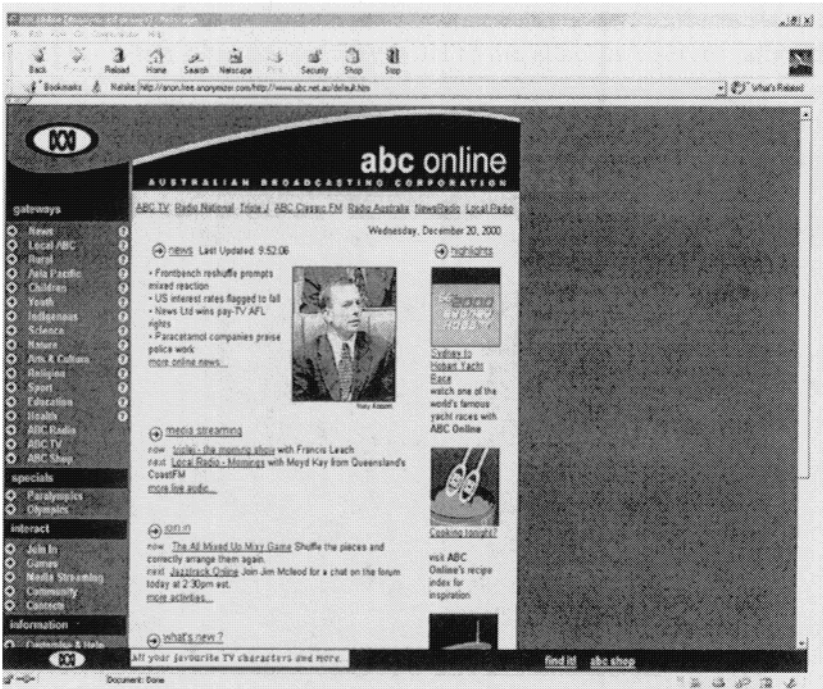


Figure 4: Real Life Example of Web Spoofing

5. BASIC DECEPTION

The basic way to mislead is to give false information as if it was true information. With web-pages the most common method is to include META Tags. These are comments lines within web-pages that represent the content of the web pages such as 'Research', 'University A', teaching or they could include META tags such as 'Pokemon', 'Britney Spears', or 'Buffy the Vampire Hunter'. This means that a search engine would wrongly list a web-page as being about a particular subject when in fact it is not. The majority of search engines rank web-sites by sending out a program called a spider, to inspect a particular site. The spider reads the META tags, determines the relevance of the web-pages information and keywords, and then ranks the site according (Deitel et al, 2001). Because of the misuse of META tags there is a growing trend among many search engines to scale down or eliminate indexing META tags (Deitel et al, 2001). Figure 5 gives an example of this where supposed information about 'Pokemon' is found in the children's pages of a white extremists web-site.

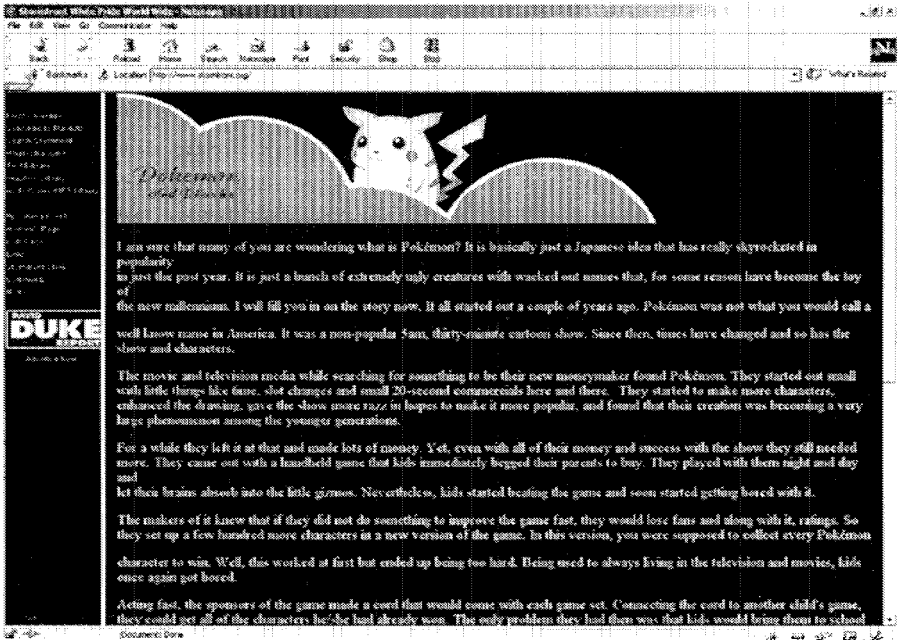


Figure 5: Pokemon Information found in a White Extremist Web-Site

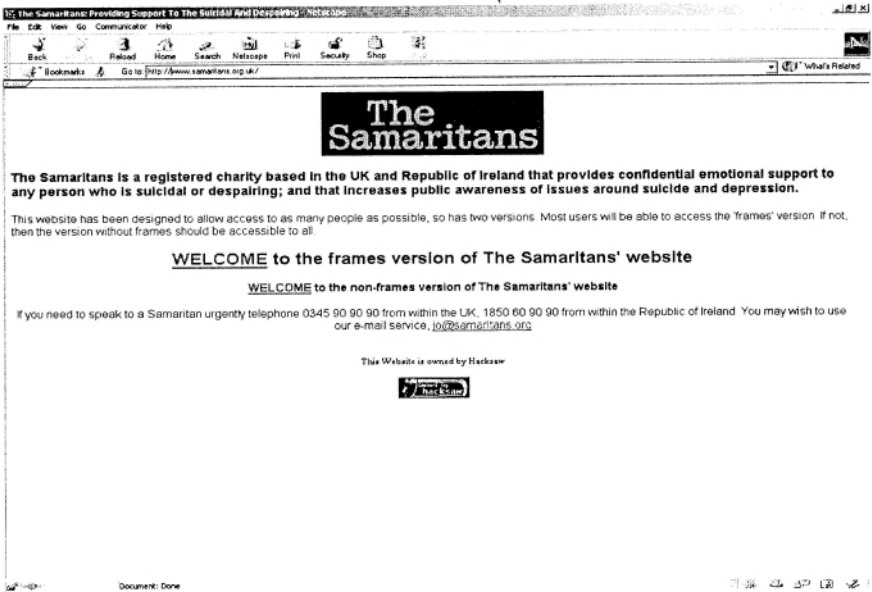


Figure 6: A successful hack of The Samaritans Web-site?

We are now facing a situation that web sites are very vulnerable to attack. Of course to deceive, attacks must not be discovered. For example, the site displayed Figure 6 has not obviously been attacked. The hack can only be identified by the *hack tag* at the bottom of the web-site. In this example the hacker has left the original web-site but only added a calling card. The hacker could easily have changed some of the context of the web-site such as the telephone number displayed or the web site links.

6. ETHICAL IMPLICATIONS

The ethical implications of deception are now becoming more important with the growth of the Internet. The advent of the Internet has expanded the amount of data available but has also decreased the reliability of much of it. As Ulfelder (1997, p.75) says: “The are no editors or safeguards to ensure that net information is fair or factual”. It is, in fact, a good medium for propoganda because “Nobody is small on the Web” (Rapaport, 1997, p.101) opportunities exist for getting viewpoints across from many. A single person with a grudge against an organisation can weave a damaging image by setting information into a specific context. Of course, organisations can do likewise. Honey pots and honey nets they can be used to capture black hats actions and method and an aid to improving security. On the other hand, the

use of web-spoofing, misuse of META tags, and alteration of web-pages could result in a situation that the content of web-pages could not be trusted because there is no assurance that the information is true or false.

7. CONCLUSION

It is interesting that in a recent survey of Australian IT managers (Hutchinson and Warren, 1999), 66% did not think there was any threat from competitors attacking their systems in any way. This perception does not bode well for the detection of acts of deception. Data integrity is an extremely important component of information management, but it must not just concentrate on internal processes of access and amendment rights. Strategies to cope with deliberate and organised attacks of a subtle nature need to be developed. Deception is one of these strategies.

8. REFERENCES

- Barry, A.M.S. (1997). *Visual Intelligence*, State University of New York Press.
- Boisot, M.H. (1998) *Knowledge Assets*. Oxford University Press, Oxford.
- Bowyer, J.B. (1982). *Cheating*, St.Martin's Press, New York.
- Brugioni, D.A. (1999) *Photo Fakery: The History and Techniques of Photographic Deception and Manipulation*, Brasseys Inc., Dulles, Virginia.
- Cohen, F (2000). *Deception Tool – kit*, URL: <http://all.net/dtk/dtk.html>
- Deitel H, Deitel P and Nieto T (2001) *e-Business and e-Commerce: How to Program*, Prentic Hall, New Jersey.
- Gerwehr, S.,Glenn, R.S. (2000) *The Art of Darkness: Deception and Urban Operations*, Rand, Santa Monica.
- Hutchinson, W.E., Warren, M.J. (2000a) The use of Deception in Systems, *Proceedings of International Conference on Systems Thinking in Management*, eds; G. Altmann, J.Lamp, P.Love, P.Mandal, R. Smith, M.Warren. 8-10 Nov, 2000, Deakin University, Geelong. pp.263-268.
- Hutchinson, W and Warren, M.J. (2000b) Deception and the Information Security function, *Proceedings of INC 2000 Second International Network Conference*, 3-6 July 2000, Plymouth, UK. pp.273-280.

Hutchinson, W., Warren, M. (1999). *The attitude and practice of Australian Information Technology managers toward Cyber-Vigilantism*, InfoWarCon99, Washington, USA, September, 1999.

Felten, E, Balfanz D, Dean D and Wallach, D (1997). *Web Spoofing: An Internet Con Game, Technical Report 540-97, Princeton University and also presented at 20th National Information Systems Security Conference, Baltimore. USA, October, 1999.*

Network Associates (2000) *Cybercop Sting* . URL:
http://www.nai.com/asp_set/products/tns/ccsting_intro

Rapaport, R. (1997). PR finds a new cool tool, *Forbes*, Oct 6, 1997, p. 101-108.

Roberts, P., Webber, J. (1999). Visual Truth in the Digital Age: Towards a protocol for Image Ethics, *Australian Computer Journal*, **31**:3; 78-82.

Spitzner, L. (2000a) *To Build a Honeypot*. URL: <http://project.honeynet.org/papers/honeypot/>

Spitzner, L. (2000b). *Know Your Enemy: A Forensic Analysis*, URL:
<http://www.securityfocus.com/ih/articles/foranalysis.html>

Spitzner, L (2000c) *To Build a Honeynet, FIRST (Forum of Incident Response & Security Teams) Conference 2000, Chicago, USA.*

This Page Intentionally Left Blank

A METHODOLOGY TO DETECT TEMPORAL REGULARITIES IN USER BEHAVIOR FOR ANOMALY DETECTION

Alexandr Seleznyov

Computer Science and Information Systems Department

University of Jyväskylä

P.O. Box35, FIN-40351, Jyväskylä, Finland

alexandr@jytko.jyu.fi

Abstract

Network security, and intrusion detection in particular, represents an area of increased interest in security community over last several years. However, the majority of work in this area has been concentrated upon implementation of misuse detection systems for intrusion patterns monitoring among network traffic. In anomaly detection the classification was mainly based on statistical or sequential analysis of data often neglecting temporal events' information as well as existing relations between them. In this paper we consider an anomaly detection problem as one of classification of user behavior in terms of incoming multiple discrete sequences. We present an approach that allows creating and maintaining user behavior profiles relying not only on sequential information but taking into account temporal features, such as events' lengths and possible relations between them. We define a user profile as a number of predefined classes of actions with accumulated temporal statistics for every class, and matrix of possible relations between classes.

Keywords: Network Security, Intrusion Detection, Anomaly Detection, Online Learning, User Profiling, User Recognition

1. INTRODUCTION

Our society is becoming increasingly dependent on the rapid access and management of information. More information is being stored and processed

on network-based computers. Increased connectivity not only provides access to larger and varied resources of data more quickly than ever before, it also provides an access path to the data from virtually anywhere on the network (Power, 1995). Thus, there is a need to have means to protect computer systems against abuse.

There are intrusion prevention and detection techniques used to protect computer systems. The intrusion prevention techniques such as authentication and authorization, safe programming, and information protection serve as a first line of defense in computer systems. However, recently the amount of successful intrusion incidents has grown quite high: even 99% of all major companies have reported at least one major intrusion incident (Sundaram, 1998). Computer systems tend to be more and more complicated introducing new weak points that allows to exploit them in order to penetrate systems' defenses. Hardware or software failures, incorrect system administration increase intrusion's chances to be successful. Software bugs also represent a great danger, since software designers are not learning from past mistakes, still reproducing "classical" programming mistakes (such as buffer overflow in *sendmail* (Sendmail, 2000)). In many cases, the security controls themselves introduce weaknesses. Thus, it shows that the usage of intrusion prevention alone is not sufficient to reliably defend computer system and there is a strong need to have another line of defense, such as intrusion detection.

Intrusion detection is a security technology that attempts to reveal and isolate intrusions against computer systems; therefore it is an important component of security system. Intrusion detection systems (IDSs) use a number of generic methods for monitoring of vulnerabilities' exploitation. They are useful not only in detecting successful breaches of security, but also in monitoring attempts to breach security, which provides important information for timely countermeasures. Thus, IDSs are useful even when a computer system has a high degree of confidence (Kumar, 1995). The intrusion detection approaches may be roughly divided into two main categories: misuse and anomaly detection systems (Smaha, 1993).

Misuse intrusion detection systems, for example (Kumar and Spafford, 1995) and STAT (Ilgun and Kemmerer, 1995), detect intrusions that follow well-known patterns of attack (or signatures) that exploit known software vulnerabilities. These misuse intrusion detection systems include encoded knowledge about poor or unacceptable behavior and directly search for it (Smaha, 1993).

The intrusion detection systems of the second category (for example IDES (Lunt et al., 1992)) are detecting abnormal behavior or use of computer resources. They classify usual or acceptable behavior and report other irregular behavior as potentially intrusive. Techniques used in anomaly detection are varied. Some of them rely mainly on statistical approaches and result in systems that have been used and tested extensively. Techniques based on prediction of

future patterns of behavior utilizing already gathered patterns is an examples of approaches tried in intrusion detection (Lane and Brodley, 1998).

In this paper we formulate an anomaly detection problem as one of user behavior classification in terms of incoming multiple discrete sequences. Although, here we focus on user-oriented anomaly detection. Monitoring multiple streams of discrete events, such as GUI events, system call traces, keystrokes, a system learns in order to classify (or recognize) user according to his behavior. By developing our approach we aim to eliminate, as much as possible, manual and ad hoc elements from the creation and manipulation of the user profiles by introducing online learning. We develop an approach that allows creating and maintaining users' behavior profiles relying not only on sequential event information but taking into account events' lengths and possible relations between them. Information about user "normal" behavior is accumulated in user profile. We define it as a number of predefined classes of actions with accumulated temporal statistics for every class, and matrix of possible relations between classes. Every class contains a number of instances, i.e. a number of patterns that are allowed for this class. In other words, an instance of a certain class contains temporal information that is peculiar for a certain pattern. A relation matrix describing possible relations between classes gives us possibility not only to check the "normality" of each action in incoming sequence of events, but also to check whether current relations between actions are "normal" for a certain user.

In this paper we develop an approach to the problem of anomaly detection. In particular, automatically matching encoded patterns against current event streams in order to find deviations from normal behavior; and than decide whether it intrusion or normal user behavior changes. Our approach is based on the assumption that the user's behavior includes regularities that can be detected and coded as a number of patterns. The information derived from these patterns could be used to detect the abnormal behavior and to learn the intrusion detection system. It is a hitherto untried approach in the field of anomaly detection.

A definition and description of temporal-probabilistic trees are given in Section 2. In Section 3 we present an approach aimed to differentiate normal behavior from abnormal by monitoring deviations between incoming events and stored in profile. Finally, in Section 4 conclusions to this work are given.

2. CLASS APPROACH TO USER PROFILE BUILDING

Traditionally, in anomaly detection, user profiles have been built by calculating statistics for different characteristics, such as consumed resources, command count, typing rate, command sequences, etc. (Lane and Brodley, 1998). In our approach we construct a user profile by defining *classes* or *cases*,

which are used as a bricks in constructing a model of user behavior, and analyze information inside classes basing on its context. We present an incoming information as a set of temporal intervals that gives us possibility to apply Allen's algebra (Allen, 1983) to discover relations between temporal intervals and to store them for further classification. In the following subsections we describe definitions and basic concepts of our approach.

2.1. DEFINITIONS AND BASIC CONCEPTS

In this section we provide necessary definitions and describe basic concepts of our approach. Here we consider a problem of user profile building as a bringing to conformity events, provided by operating system log facilities, with notions used to build a user behavioral model. Thus, the system may possibly have several streams of discrete events such as GUI events, system call traces, network packets, and keystrokes. It needs to automatically build a profile for every user in order to recognize him in a future fitting current behavior into behavioral model described in his profile. During this process, the system should use as less as possible manual and ad hoc elements. In other words our aim is not only to develop an approach for behavioral model description, but also to automate all processes used for creation and manipulation of the user profiles as much as possible.

At the beginning we introduce *a layer structure* of events (Seleznyov and Puuronen, 1999a), which is a three levels used for describing incoming information on different abstraction levels.

The term *event*¹ has been widely used within the temporal database area giving it different meanings. We define the event *as a single indivisible occurrence on the time axis*. As can be seen from this definition that the type of a possible occurrence is not defined. Therefore, we may conclude that the event meaning may depend on source of the incoming information. In other words, applying the concept of event on different types of source information we are going to have different results. For example, for a GUI log an event may be represented by a GUI message, for network packets it may be a header of a single packet, etc. Applying our definition to the examples we can conclude that in these cases a single record in a log file represents an event.

In order to describe an information abstraction way we define a notion of layer that reflects different levels of information generalization. The higher layer the more general and more descriptive the notions describing the user behavior are. Thus, on the highest layer we describe the user behavior using most general way not depending on a source where information is coming from.

In this work we are using an underlying assumption that a person's interaction with a computer consists of different activities that he performs in order to achieve his goals. These activities consist of actions. Each action causes

series of events in the operation system. Each user performs similar activities which are expressed by repeated sets of actions and which differ on a per-user basis. This gives the possibility to differentiate an intruder from a valid user (Seleznyov and Puuronen, 1999a).

Layer is a concept generalization level of relations between occurrences, each of which represents a single event on a different abstraction level (Seleznyov and Puuronen, 1999b). At the lowest instant layer all occurrences are represented as a time points (instants) on an underlying time axis. A single occurrence on this layer is called an event. It is equivalent to a single line in the audit trail. An event is described by a single instant relative to a particular user. Information on this layer is source-dependent (for example, it depends on operating system or logging facility used to collect it). Thus, same occurrence may be defined differently on this layer.

On the *interval* or *action* layer events with their simple relations are described and they form actions. The action is considered as a temporal interval, as for example: LOGIN-LOGOUT. A *relation* defines a temporal relation between two events as one of Allen's point temporal relations (Allen, 1983) and has a *Name*. The difference of any two-time points is likewise a rational number (Kautz and Ladkin, 1991). There are three basic relations that are used to represent relations of events: $<$ - "less" relation, $=$ - "equal" relation, and $>$ - "greater" relation.

The most complicated level is the *activity layer*, which is represented by actions and relations between them. It describes them in a general source-independent way. Because the actions are extended in time, different actions may overlap in time and interact. One LOGIN-LOGOUT temporal interval, for instance, includes dozens of mail check intervals. A single occurrence on this level we call an *activity*. A *Relation* between two actions (temporal intervals) is defined as one of Allen's interval temporal relations (Allen, 1983).

These three layers are the way by which systems classify certain patterns of change. No one is more correct than other, although some may be more informative for certain circumstances. They are aimed to manage incoming information from multiple sources. For example, if system detects that a WWW browser is active and it exchanges information using HTTP protocol then it may conclude that user is browsing WWW pages in Internet. If the system observes network packets' headers it may come to the same conclusion when it detects connection establishment between user workstation and some server on port 80 followed by an information exchange. It may come to a same conclusion when observing some sequence of system messages between different modules of an operating system. Therefore, our point is - by monitoring different sources (sequences of events) it is possible to come to the same conclusions or, in other words, build a string of user activities, which are source and platform independent. And layer structure is aimed to make this transformation from event

to activity layer, making possible to combine multiple strings from different sources. Moreover, wide usage of I/O caching introduced some uncertainty in determining exact time points for a certain events. System *read* and *write* operations may be delayed and appear later in system logs than the user actually performed them. Getting confirmations from different sources the system is able to reason about actual time the user has requested a certain operation².

2.2. USER PROFILE

Above we have described a possible structure of information that may be gathered from operation system. This structure is used for abstracting information obtained from audit log files to more general - source or platform independent, giving it more meaning in context of person-computer interaction. How to store and process the obtained information? Below we present a way to construct user profiles using information obtained from different sources.

In contrast to definitions related to the structure of information layers, where all notions were defined in a way that more general were given in terms of more specific. For a user profile description we are going to move in a reverse direction. Hence, the structure of user profile definitions resembles an inheritance mechanism in object-oriented programming languages. Thus, we go from general to more specific, where more specific definition is formed by inheriting all features of "parent" one and adding some additional value to it.

As a general concept we define *action class* which describes one of the possible kind of action. It provides a formal description of an action without providing any specific details. Action class contains descriptions of events that start and end that action and possible events between them. Continuing our previous example consider a hypothetic action class "Web browsing". It may be defined by Web browser activity interval. Also if we monitor network packets we may define same action class by a long sequence of events. For example, a request for connection establishment between some server and a user workstation port 80 (handshake protocol) followed by some information exchange and closing connection. As a matter of fact a number of all possible actions is limited by operation system tools and additionally installed programs, therefore, a number of action classes is finite and known beforehand.

An *action class instance* is an instance that describes a certain group of actions that belong to the same action class and have similar temporal characteristics. By similar temporal characteristics we imply temporal distances (time lengths) that characterize actions. These distances must be distributed normally in order to be grouped into a same instance. In figure 3.a it is possible to see an example of an action class that contains three instances. These instances described by three normal distributions each of which has own parameters - μ and σ . Finally, every instance contains n - number of actions grouped in it.

It is used for calculation of probability distribution of instances inside a single action class.

As was mentioned before an action class instance is formed by a number of *actions* with similar temporal parameters. Every action has information to which action class it belongs and it represents a concrete happening. Since an action has a beginning and an ending it is described by a temporal interval that has length (action's length).

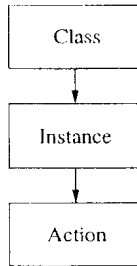


Figure 1 Structure of information stored in user profiles.

In this section we presented a way to represent information about user actions taking into account their temporal parameters. It describes user events by forming actions, classifying them, and splitting them into instances of the same action class. In figure 1 it is possible to see a structure that describes a part of a user profile presented in this section.

2.3. RELATIONS BETWEEN ACTION CLASSES

Looking at previous work we may summarize that traditionally, in anomaly detection, user profiles have been built basing on different characteristics, such as consumed resources, command count, typing rate, command sequences, etc. In these cases information analysis has been made using system log files, command traces, and audit trails. In most cases the classification was based on the sequence of events in time. However, the sequential data is not the only information that is possible to get from a stream of discrete happenings. It also contains some hidden information that is usually neglected: time relations between events are not taken into account at all or only very little attention is given to it. Sometimes time relations play a crucial role in attempting to classify events, i.e. determining whether an event is a part of anomalous or normal behavior. For instance, if an account has been under an IP spoof attack (Phrack, 1996), it is easier to recognize it relying on time relations between events, since the misuse activity appears as a continuation of the normal behavior

within a single session. To be able to expose and later use of relations between actions it is necessary to introduce additional concepts.

Relation class is a notion that describes one of all possible relations between any two actions: $Action_i$ and $Action_j$. It is defined as one of Allen's interval temporal relations (Allen, 1983) and it has a *Name*.

$$Name \in \{before, after, meets, met - by, during, includes, overlaps, overlapped - by, starts, started - by, finishes, finished - by, equals\} \quad (1)$$

According to (Allen, 1983):

- given any interval, there exists another interval related to it by each of the thirteen relationships;
- the relationships are mutually exclusive;
- the relations have a transitive behavior, e.g. if A is "before" B, and B "meets" C then A is "before" C.

Relation class instance describes some set of relations that have similar temporal characteristics and each of them belongs to the same relation class. In other words, a relation instance has a *Name* and describes some distribution of temporal parameters of relations grouped by this instance.

Relation is a relation (one of thirteen presented above) between two any actions $Action_i$ and $Action_j$. It is characterized by a *name* and a temporal distance between these actions. Thus, name is a qualitative parameter that describes what kind of relation it is, and temporal distance is a quantitative parameter that shows how strong the relation is or how much of it is possible to find between two current actions.

What is a temporal distance in context of user behavior expressed by a sequence of actions? Below we consider all possible relations between actions and define a notion of temporal distance for them. There are thirteen possible relationships between two actions (Allen, 1983). In our approach we are not using all of them. Since we have qualitative temporal characteristics we may define several *basic relations*, which with different temporal parameters produce all possible relations. For example, if $A1 : before A2$ and temporal distance (distance between end of $A1$ and beginning of $A2$) is zero then $A1 : meets A2$.

We define two relations as *basic*: "before", "during". Figure 2 shows them. Below we define temporal distances t for basic relations.

$$\forall A1 : before A2, t(A1 : before A2) = A2_begin - A1_end \quad (2)$$

If $t(A1 : before A2) = 0$ then we have case when $A1 : meets A2$. If $t(A1 : before A2) < 0$ in terms of Allen's temporal relations we may say that $A1 : overlaps A2$.

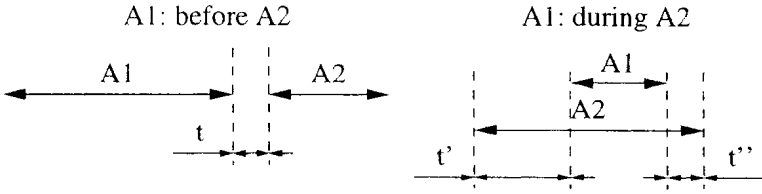


Figure 2 Basic relations.

$$\forall A1 : \textit{during} A2, t(A1 : \textit{during} A2) = A2_begin - A1_begin \quad (3)$$

As it is possible to see from figure 2 a position of the interval A1 relatively to A2 is defined by t' and t'' . If $t' = 0$ then this basic relation forms A1 : starts A2 relation. In case when $t'' = 0$ we have A1 : finishes A2. When both $t' = 0$ and $t'' = 0$ then A1 : equals A2. For our purposes we do not need to calculate and store t'' . Since we have a relation Name, t' and A1 length we are always able to reconstruct t'' when we need it. That is why we defined the temporal distance for a "during" relation as t' .

As we can see our two basic relations include all seven Allen's relations³. Therefore, in order to create a user profile we transform incoming information into vector of N classes with instances inside them. To take into account important relations between actions we need to discover them and store their temporal characteristics.

To represent relations between actions we use a square matrix $N \times N$ - relational matrix. In every cell $\{i, j\}$, matrix holds relational classes that are allowed between two classes i and j . Thus, cells i, j when $j > i$ contain direct relations for A_i and A_j , and cells i, j when $j < i$ contain reverse ones.

Using the relational matrix we may check whether there is a certain relation between any of two classes and if it fits into a certain relation instance.

3. DETECTING ABNORMAL BEHAVIOR

In this section we are going to discuss how to use described above action and relation classes to discover abnormal behavior by monitoring deviations between current user behavior and a model stored in profile. N action classes and a relational matrix are considered as tools that describes the model of user behavior. Every action class has one or more instances that represent some user action characterized by statistic parameters of time distribution - mean and standard deviation. Role of transactions' description between actions fills the relational matrix, every relation instance in which is characterized by same kind of temporal parameters. These tools are used to classify user behavior. There-

fore, deviations from current values of its sequential and temporal parameters are considered as a consequence of abnormal behavior. To estimate the value of deviation we introduce *coefficient of reliability* - $r = \overline{[0, 1]}$. It is assigned to every active user and shows probability that the user is someone who he claims to be. During classification the system monitors deviations between expected or predicted user behavior and current one. According to this deviations it calculates some penalty (negative) or encouragement (positive) value, which reduces or increases the coefficient by Δ_r . At a root node the coefficient is assigned to 0.5 since there is no yet any evidence neither of distrust nor of trust. Therefore, the coefficient of reliability has an ability to grow as well as diminish. If it crosses a certain threshold it means that there is a sequence of actions where parameters of each action are not in admissible intervals. It is considered as a case of abnormal behavior and thus, an alarm should be fired.

Coefficient Δ_r is calculated at the each step of classification. Each time the system gets a new case for classification it needs to determine correct action class and instance. Finding a correct class is relatively easy. Knowledge an action's name points to a certain class. After this it is necessary to find a correct instance inside the class. Below we present our way to automatically determine the instance, where the current action belongs, among several inside one class:

- 1 first it is necessary to determine a distribution where a new action belongs - $t \in [\mu - 2\sigma; \mu + 2\sigma]$;
- 2 if there are more then one interval found on step one, then we find $\min |\mu - t|$ among them;
- 3 if there are more then one instance with same temporal distances between them and a current action, we chose the one with smallest standard deviation: $\min [\sigma]$.

In normal distribution 95% of all cases are lying in interval $[\mu - 2\sigma; \mu + 2\sigma]$. During first step we use this to determine where a new case belongs. If there are some action instances that are overlapping each other and the new case is in overlapping area we use step two and three to introduce additional restrictions to find a right instance for the new case.

Below we consider calculations of the temporal-probabilistic characteristics for an action instance. In order to save computer resources a system does not need to keep a history of events, it needs only to have two parameters that describe a distribution inside this instance. Therefore, if a new case comes that supports a current instance we update its temporal parameters. For every i node its mean and standard deviation calculated each time it being used for classification:

$$\mu = \frac{\mu \times (n - 1) + t}{n} \quad (4)$$

where n - number of cases in this instance;
 t - temporal length of being classified action.

$$\sigma = \sqrt{\frac{\sigma^2 \times (n - 1) + (\mu - t)^2}{n^2}} \tag{5}$$

The same formulas may be applied for calculations of temporal parameters of relation instances. In this case t is a temporal length of being classified transition.

How we may use these temporal characteristics of action and relation instances to detect abnormal behavior? Well, if we take one instance it contains some distribution described by μ and σ . We separate all area outlined by this curve into three areas:

- Area limited by one σ - 68% of all area. In case when a new case is in this area we consider this case as a strong match that supports this instance. Therefore, the coefficient of reliability should be "encouraged".
- Area between one and two σ - 27% of all area. When a new case gets into this area it is a weak match. In other words, we have proofs that the new case belongs to this instance, but it does not support it as much as in previous case. Thus, the coefficient of reliability should not be changed.
- Area beyond two σ . In this situation the value of penalty for the coefficient of reliability should be calculated.

Below we present a formula for calculations of the coefficient of reliability changes:

$$\Delta_r = \begin{cases} 0 < t \leq \mu - 2\sigma, \nu_i \times \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) - \exp(-2) \\ \mu - 2\sigma < t \leq \mu - \sigma, 0 \\ \mu - \sigma < t < \mu + \sigma, \frac{n}{c} \times (\exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) - \exp(-0.5)) \\ \mu + \sigma \leq t < \mu + 2\sigma, 0 \\ \mu + 2\sigma \leq t < \infty, \nu_i \times \exp\left(\frac{-(x-\mu)^2}{2\sigma^2}\right) - \exp(-2) \end{cases} \tag{6}$$

where c - coefficient that limits n and determines system's sensitivity to deviations;

ν_i - coefficient of security significance of the action (in other words, how much danger improper usage of this action may cause); it is defined by system administrator for a certain kind of action.

Coefficient of reliability change calculations are based on following assumptions:

- how big difference is between predicted and current action lengths;

- how big difference is between predicted and current transition lengths;
- security significance of the current action;
- how big standard deviation a current action instance has;
- how big standard deviation a current relation transition has.

At the end of classification process if a coefficient of reliability r is lower than a certain threshold an alarm is fired. The alarm may also be issued during classification process when the coefficient of reliability diminishes very fast below the threshold. In the figure 3 we may see an example of some class "X" and a graphical representation of distribution of coefficient of reliability changes inside this class.

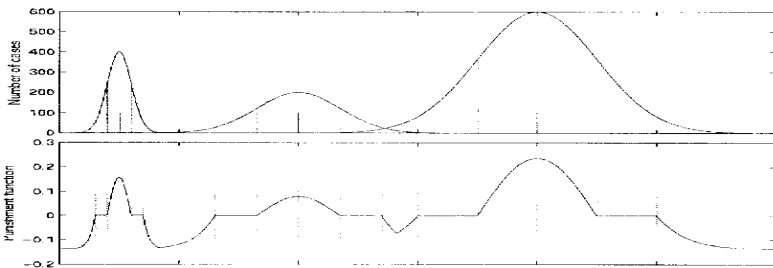


Figure 3 Class "X": a) three instances of the action class; b) dynamic Δ_r changes inside this action class.

In this section we presented an approach to expose regularities in user behavior. Using these regularities a user profile is built to monitor user's current behavior and detect anomalies in it. Basing on the amount of these anomalies it is possible to decide whether the current user is the same user he claims to be.

4. CONCLUSIONS

In this paper we proposed theoretical background for a new approach for anomaly detection. This approach allows to create a profile for every user by

automatically finding regularities in his behavior and then constantly update these profiles. The main assumption behind this approach is that the behavior of each user follows regularities that may be discovered and presented using a limited number of action and relation classes.

The profiles are created by forming a vector of N action classes each of which contains several instances. In other words, it contains several temporal patterns of some action. Every profile also contains a matrix of relation classes (they are similar to action classes but describe relations). This matrix allows us to check whether a relation is valid between every two action classes.

Using described profiles a monitoring system evaluates every user action according to its length and relations with previous and next actions. During classifications a coefficient of reliability is changed. Basing on it a decision is made whether the current behavior is normal or anomalous.

The presented in this paper approach has some advantages. It is relatively simple and easy to visualize. It is quite fast since it does not require many calculations. The user profile is represented by action and relation classes, which are described by same temporal parameters, and thus, it is possible to use same formulas to calculate and update actions' temporal parameters as well as relations'.

At every time point every class contains several instances and therefore, may be described by some curve (as in figure 3a). To eliminate some calculations it is possible not to describe every instance as a distribution of temporal parameters, but a distribution of a coefficient of reliability change value (as in figure 3b). It would not require calculations of this value at every step and thus, increases classification speed.

The approach presented in this paper is interesting from theoretical point of view. However, we are still in the initial stages of our research and further development, numerous tests and evaluations with real implementation are needed to verify the practical aspects of this approach.

Notes

1. In this paper we attempt to use all definitions and terms in accordance with (?) and (?).
2. Since we are using a user-oriented approach we are interesting in time when a user has requested a certain action not when it actually has been performed by operation system.
3. Six reverse relations we do not take into account. Later we explain reasons for this.

References

- Allen, J. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26:832–843.

- Ilgun, K. and Kemmerer, R. (1995). State transition analysis: A rule-based intrusion detection approach. *IEEE Transactions on Software Engineering*, 21(3):181–199.
- Kautz, H. and Ladkin, P. (1991). Integrating metric and qualitative temporal reasoning. In *Nine National Conference of Artificial Intelligence*, CA, USA.
- Kumar, S. (1995). *Classification and Detection of Computer Intrusions*. Phd, Purdue University.
- Kumar, S. and Spafford, E. (1995). A software architecture to support misuse intrusion detection. In *The 18th National Information Security Conference*, pages 194–204.
- Lane, T. and Brodley, C. (1998). Sequence matching and learning in anomaly detection for computer security.
- Lunt, T., Tamaru, A., Gilham, F., Jagannathan, R., Neumann, P., Javitz, H., Valdes, A., and Garvey, T. (1992). A real-time intrusion detection expert system (ides) - final technical report. Technical, Computer Science Laboratory, SRI International.
- Phrack (1996). Ip-spoofing demystified: Trust-relationship exploitation. *Phrack Magazine*, available from <http://www.fc.net/phrack/files/p48/>, 7(48).
- Power, R. (1995). Current and future danger. Computer Security Institute, San Francisco, California.
- Seleznyov, A. and Puuronen, S. (1999a). Anomaly intrusion detection systems: Handling temporal relations between events. In *2nd International Workshop on Recent Advances in Intrusion Detection*, Lafayette, Indiana, USA.
- Seleznyov, A. and Puuronen, S. (1999b). Temporal aspects of user profiling in anomaly detection. In *Fourteen International Symposium on Computer and Information Sciences*, Izmir, Turkey.
- Sendmail (2000). *Sendmail Mail Program*. Description and new version available from <http://www.sendmail.org>.
- Smaha, S. (1993). Tools for misuse detection. In *ISSA '93*, Crystal City, VA.
- Sundaram, A. (1998). An introduction to intrusion detection. *ACM Crossroads*.

ADELE: AN ATTACK DESCRIPTION LANGUAGE FOR KNOWLEDGE-BASED INTRUSION DETECTION

CÉDRIC MICHEL, LUDOVIC MÉ

Supelec, BP 28, 35511 Cesson Sévigné Cedex - France

{cmichel,lme}@supelec-rennes.fr

Keywords: Intrusion detection, attack description language

Abstract ADeLe is an attack description language designed to model a database of known attack scenarios. As the descriptions might contain executable attack code, it allows one to test the efficiency of given Intrusion Detection Systems (IDS). Signatures can also be extracted from the descriptions to configure a particular IDS.

1. INTRODUCTION

In this article, we introduce an attack description language, ADeLe, designed for knowledge-based intrusion detection (also called misuse detection).

The primary goal of ADeLe is to combine all the knowledge available for a given attack in one and only one readable and high-level description.

ADeLe is then to be used to model known attack scenarios in order to build an attack database. This database should then be used itself: to configure the probes and detection engines of a given intrusion detection system (IDS) or to test the detection capabilities of a given IDS (by means of attack replay).

The ADeLe language has been developed simultaneously with the Lambda [2] language within the Mirador¹ project, which may account for significant overlaps between the two languages. Both languages allow the expression of the attack code as well as rules for detection and correlation. However, Lambda uses a declarative approach while ADeLe uses a more procedural approach.

¹Project funded by the DGA/CELAR/CASSI which is part of the French Ministry of Defense.

This article is organized as follows: section 2 describes some related work and presents the main characteristics of an attack description in ADeLe. Section 3 details the language itself. Section 4 summarizes and outlines future work. Finally, an example of attack description in ADeLe is given in the appendix.

2. ADELE AND RELATED WORK

As attacks against computers and networks are becoming more frequent and more sophisticated, there is a need for representing and sharing information about these attacks [5]. A simple intrusion detection signature is far from being sufficient to describe an attack. We also need to know what the exploited vulnerability is, how the attack was performed, what are its consequences, and how to react (automatically or not) in order to stop it.

To describe an attack completely, i.e. to describe every aspect, we need to use several specific languages, with different purposes. References [17, 5] propose six different classes of such “attack languages”: exploit, event, detection, correlation, reporting, and response. Exploit languages [14, 4, 2] are used to describe the stages to be followed to perform an intrusion. Event languages [15, 8, 1] describe the format of events used during the detection process. Detection languages [10, 12, 13, 5, 11, 2] allow the expression of the manifestation of an attack in terms of occurrences of events. Correlation languages [2] permit analysis of alerts provided by several IDS in order to generate meta-alerts. Reporting languages [6, 3] describe the format of alerts produced by the IDS. Finally, response languages are used to express countermeasures to be taken after detection of an attack.

In table 1, ADeLe and the previously referenced attack languages are organized into these classes. The majority of these languages specifically address one aspect of the attack.

As we could not find any language allowing the description of all aspects of an attack, we decided to create ADeLe while Cuppens and Ortalo were creating Lambda [2]. So, ADeLe is directly concerned with four classes of attack languages: exploit, detection, correlation and response. A description in ADeLe thus contains three parts (as detection and correlation are merged). We propose neither a report language, as we made the choice to use IDMEF [3] in order to facilitate inter-operability between IDS, nor an event language, as the notation we chose to designate fields within events (cf 3.2.1) is an intermediary representation independent of the raw format.

ADeLe relies on a largely accepted intrusion detection framework, in which we assume that there are probes and detection engines. Probes are located in various places of the monitored environment delivering events. Detection engines from particular IDS (host-based or network-based) deliver alerts. We also assume that these alerts use the IDMEF format [3].

	BSM	Tcpdump	Bishop	CASL	NASL	CJSL	IDMEF	Kumar
	[15]	[8]	[1]	[14]	[4]	[6]	[3]	[10]
Event languages	x	x	x					
Exploit languages				x	x			
Reporting languages						x	x	
Detection languages								x
Correlation languages								
Response languages								

	BRO	Snort	SNP-L	STATL	G ₃ SATA	LAMBDA	ADeLe
	[12]	[13]	[16]	[5]	[11]	[2]	
Event languages							
Exploit languages						x	x
Reporting languages							
Detection languages	x	x	x	x	x	x	x
Correlation languages						x	x
Response languages							x

Table 1 Examples of specific attack languages (non exhaustive list)

ADeLe is designed to allow attack descriptions which are: readable (XML-like tags are used to encapsulate each part), comprehensive (it is possible to represent every aspect of an attack, i.e. from the attacker’s and the defender’s points of view, in only one description), generic (operators allow the exploit part and the detection part to be generic to reduce the number of attacks present in the database by having only one description for multiple variants of the same attack), and modular (defining an attack composed of several attacks already described is allowed and defining a meta-alert also).

ADeLe was proposed with the dual purpose of being able to configure and also to test IDS. Indeed, an ADeLe description is not directly operational. It contains information about attacks, but this information needs to be extracted and transformed through several compilers² (or interpreters) performing the configuration. Because an ADeLe description contains source code of the

²One compiler extracts everything defining what the interesting events are for a particular attack and transform it to configure the concerned probes. Another compiler extracts the signature of the attack scenario in terms of occurrence of events or alerts and thus allows configuration of the detection engine for a particular IDS.

attack, we are also given the opportunity to constitute a database of executable attacks. Given the appropriate interpreter (or compiler), it is possible to run a complete database of attacks against intrusion detection systems. It allows testing of the detection efficiency of those IDS in terms of false negative rate.

3. ATTACK DESCRIPTION IN ADELE

An attack description written in ADeLe is organized as presented in the appendix. References to line numbers refer to the source code of the example in the appendix.

An ADeLe description looks like a function in C with name and parameters (cf line 1). The name is the one that will be reported in the alert when detection has occurred. Possible typed parameters can follow, representing input and output variables. They are used to make the exploit part re-usable. If one describes in ADeLe a global attack scenario made of several smaller attacks, also described in ADeLe, he/she just has to reference the name of these attacks with the appropriate parameters. In most cases, input parameters refer to information needed to launch the attack, and output parameters refer to information or an access level gained by the attacker.

The body of the description is made up of three parts. We describe the EXPLOIT part in 3.1. Then we detail the DETECTION part in 3.2. Finally, we present the RESPONSE part in 3.3.

3.1. THE EXPLOIT PART

The first part of the attack description represents the attacker's point of view. It links together three aspects of the attack: the requirements for launching it, its code (or its stage description), and the results gained by the attacker. Thus, the exploit part is composed of three sub-parts: pre-condition, code, and post-condition.

The <PRECOND> section. In this section, we can express the requirements for launching the attack. Most of the time, it is knowledge about the vulnerabilities that must be present on the target. It can be specific to the target (operating system, version of installed software...). It can also be something more general, like the level of privilege (as defined in [9]) needed by the attacker to launch a successful attack against the target³ (cf line 4).

The <ATTACK> section. This is the location of the source code of the attack. We allow this code to be expressed in any language. We use a specific tag

³However, the attacker can launch a blind attack without ensuring that he has the required privileges. In this case, the attack will probably fail.

(cf line 6) to identify the language used and to allow an appropriate interpreter to execute the code of the attack. It can be a general programming language such as “C,” “C++,” “Perl,” or specific exploit languages such as “Casl”[14] or “Nasl”[4]. In the case where no source code is available, we can use an informal textual description of the attack (“Text”).

We have begun to define a high-level scripting language adapted to exploit code writing: “EDL”⁴. It relies on classical scripting languages concepts: typed data, variables, boolean expressions, mathematic operators, a few keywords to ensure flow control, and libraries of functions. Those functions can be high-level functions (e.g., commands used during a FTP connection) as well as low-level functions (e.g., network packet generators).

To describe the actions defining an attack, we use these functions (cf line 13). The actions defining the attack are executed in the order where they appear, except if there is application of an action operator.

Actions operators. In order to represent the variants of the same attack, we introduce three operators: **Non_ordered**, **One_among**, and **Subset_of**. Use of these operators allows us to reduce the database size because we describe a single, but more generic, attack.

Actions (or groups of actions) can take place within the *Non_ordered* operator (cf lines 16-29): they all have to be executed in any order. If, for a stage of the attack, a choice can be made among actions (or groups of actions) equivalent in terms of consequences, the *One_among* operator can be used (cf lines 41-50). That represents the execution of only one of the actions of the set. To express the execution of a subset of actions of cardinality C among a set of N actions ($1 \leq C \leq N$), the *Subset_of* operator can be used. The execution order is unspecified.

For execution, various strategies will have to be considered for the choice of the variants of the attack (e.g., random choice of action, always the first possibility...).

Types and operators. We allow the declaration of variables (of type String, Integer, Boolean IPAddr...). We also define classical boolean operators (logical operators, comparisons...) and pattern matching in the character strings (keyword IN).

Conditional blocks. The actions defined in the attack scenario produce intermediate results which must first be tested to decide on the continuation of the attack. It is defined as follows:

```
IF (<boolean expression>){ <actions> } ELSE { <actions> }
```

⁴Exploit Description Language.

Iteration. To describe the repetition of actions, the “WHILE” iteration can be used: `WHILE (<boolean expression>){ <actions> }`

A link between attack and detection.

The role of the `<DETECT>` part is to give indications about the way the attack can be detected. As the events (or alerts) correspond to stages of the attack, it is useful to establish a link between the actions and their detection. This is why, at the time of the expression of the attack scenario, we associate events/alerts with the actions or groups of actions observable by an IDS. In concrete terms, we delimit the portion of the attack scenario which corresponds to a type of event/alert in order to be able to name it for later reference in the `<DETECT>` part (cf lines 12-14). If we are unable, for a given attack, to observe its intermediate stages, we will define only one event that includes all the stages of the attack. In certain cases, an action is observable (detectable) neither on the target network nor on the target system (e.g., the attacker is cracking passwords on his/her local machine). It will then be associated with no event.

The `<POSTCOND>` section. In this section, what has been obtained by the attacker is expressed. Most of the time, it is an increased level of privilege (cf line 62). It could also be disclosure of information, corruption of information, denial of service, or theft of resources [7]. Moreover, as proposed by a reviewer, these last goals could be refined in order to express more specific gains, which could be subgoals in chain of attacks. For example, we could add changes in configuration, impersonation, injection of false data, creation of a backdoor, etc.

3.2. THE DETECTION PART

In this part, we propose a new high-level language to express the detection of attacks. This language allow us to write basic signatures as well as complex scenarios involving combination of known attacks. It is made possible because events coming from the probes are handled in the same way as alerts from IDS.

There are three sections related to the detection. Section 3.2.1 details how the detection itself is expressed. Section 3.2.2 deals with the confirmation of the diagnosis of detection. And finally, Section 3.2.3 describes the emission of an alert using the IDMEF format.

3.2.1 The `<DETECT>` section. This section is itself divided into three subsections. The first one is used to name the events/alerts expected to appear during the attack and to define their general types. In the second one, we express the temporal correlation between those named events/alerts. In the last one, we refine the characteristics of each event/alert and also the contextual correlation between events/alerts belonging to the same attack.

The <EVENTS> subsection. A name is given to each event/alert that should be observable during the attack. At this stage of the description, we only define the general types of those events/alerts. If possible, the names should already be defined in the <ATTACK> section (cf 3.1).

We decided to use the IDMEF format [3] **notation** not only for alerts returned by IDS, but also for events coming from the various probes. The prefix for alerts and events is then one of the following: “Alert.,” “Network.,” “System.,” or “Appli.” (cf lines 69-75).

However, as the IDMEF data model is designed only for alerts, we use a similar hierarchical notation to allow access to the fields of events from the probes, as explained in the <CONTEXT> subsection.

The <ENCHAIN> subsection. Once the events⁵ are named, we express their temporal relationships (except when the detection of the attack involved a single event/alert). In this section, we define the **global scenario** of detection. It can be seen as a temporal automaton in which transitions are fired when expected events occur. The detection is achieved when the last event specified in the scenario has occurred.

To describe the global scenario of detection (in only one expression), we use any combination of the following operators (cf line 78):

- **sequence:** two events that should appear in sequence are separated by a semicolon (whatever may be the number of events of different types occurring in-between). For instance, the condition $(E0 ; E1)$ is verified for the following events flow: $(E0 A B E1)$.
- **unspecified order:** when several events (or groups of events) must all occur in the attack in an unspecified order, the operator `Non_ordered{<events>}` can be used. It defines a temporal interval for which the boundaries are the occurrences of the first and the last events.
- **exclusive choice:** to express that one among several events can be used for one stage of the attack, we use the operator `One_among{<events>}`. The condition is verified as soon as one (and only one) event of the set has occurred.
- **subset:** to express that one or more events from a set should occur (in no particular order), we use the operator `Subset_of{<events>}`.
- **repetition:** to express a series of n events of the same type, we use the notation: $E1 \wedge n$.
- **aliases:** in the <EVENTS> subsection, we manipulate only occurrences of general events. As soon as an event appears several times in the scenario,

⁵ In this subsection, we use the term events to designate both events and alerts.

we cannot isolate a particular occurrence of this event. For instance, $E_0 ; E_1 ; E_0$ represents two different occurrences of E_0 . In order to isolate a particular occurrence, we allow the declaration and assignment of a variable. For example, to isolate the 7th occurrence of 10 similar events, we write: $(E_1)^{\wedge 6} ; EVT1 := E_1 ; (E_1)^{\wedge 3}$. This expression can be considered as an alias definition for a particular event.

- **non occurrence:** to allow an IDS to invalidate the hypothesis that the attack is in progress, we define an operator to express the fact that a particular event should not occur during a specified interval of events. Example: $\{E_0 ; E_1 ; E_2\}$ WITHOUT F . This example means that the event F should not occur between events E_0 and E_2 .
- **time constraints:** in addition to the global scenario, we can express time constraints⁶ between events in order to invalidate the hypothesis that the attack is still in progress. If we define multiple constraints, they simultaneously apply to the detection scenario.

Example: $(E_2. Time. time - E_1. Time. time) \leq "2mn"$

The <CONTEXT> subsection.

For readability's sake, we chose to separate the temporal (<ENCHAIN>) and the contextual (<CONTEXT>) aspects of a scenario.

A context rule is a constraint applied to **occurrences** of one or more previously defined events/alerts (or aliases). All the constraints must be satisfied simultaneously⁷.

Constraints can be expressed on any field to filter only events/alerts that should appear for a given attack. Some constraints apply on only one event; they are used for configuration of the concerned probes. Other constraints link up several events/alerts which belong to the same attack⁸; they are used to configure the detection engines for a given IDS.

When alerts are linked by constraints, it is called **alert correlation**.

Notation of fields within events.

As said before, we designate the fields of events using the IDMEF notation whenever it is possible. Each kind of event (prefixed with "Network," "System," or "Appli")⁹ should have **at least** the following fields:

- **alertid:** serial number for the event
- **Time.*:** time of the generation of the event

⁶ Most of the time, it is an explicit timeout.

⁷ Agreed that a constraint is only applicable when the concerned event/alert has occurred.

⁸ For instance, all events refer to the same target.

⁹ Fields within alerts are prefixed with "Alert". Their notation is fully defined in [3].

- **Analyzer.id**: identifier for the probe which has produced the event
- **Classification[0].name**: name used to refer to that kind of event
- **Target[0].***: fields containing information about the target of the event
- **Source[0].***: fields containing information about the source of the event

However, because IDMEF was designed for alert purposes, we needed to add specific fields for each kind of information source. Those additional fields allow us to access low-level information contained in events.

The notation we use to access the fields of instantiated events is easily extensible if necessary. It is a high-level view of the events. It allows us to hide the details of the raw format.

For example, to express interesting network events, we propose the following hierarchical object notation for the fields (non exhaustive):

```
Network.Ip.Header.(version|tos|ttl|src|dst|...)
Network.Ip.Tcp.Header.(src|dst|seqnum|flags|...)
```

As system events are more platform-dependent, we define fields such as:

```
System.Bsm.(event|program|args|env|...)
System.Nt.(...)
```

Events from application logs are defined in the same way:

```
Appli.Apache.(...)
Appli.Wuftp.(...)
```

Following this notation for alerts and events, here are some examples of constraints:

- comparison (`==`, `<`, `>`, `<=`, `>=`, `!=`) between a numeric field and a constant (cf lines 81-82)
- one or more fields unified with a variable so that any new occurrence of this variable represents the same value (cf lines 83-89)
- numeric field value included in a list of values or intervals (keyword `IN`)
- locating substrings expressed by regular expressions in a given field (keyword `MATCHES`)
- alternatives in a constraint (logical `OR`)

3.2.2 The <CONFIRM> section. In this section, we express what to do to verify that the attack, if successful, has produced the expected consequences. This is a way to eliminate some false positives. To achieve this goal, we define a few boolean functions¹⁰:

¹⁰This list is not exhaustive.

`Wrong_Hashes (<file name>)`: evaluates to `true` when the file has been tampered with. It can be used for a Trojan Horse attack, in order to compare the digests of the binaries (which are suspected to be corrupted) with the last safe version.

`File_Contains (<file name>, <regular expression>)`: evaluates to `true` when the contents of the file match the given expression. It can be used to test the contents of a file (possibly modified).

`Unreachable_Machine (<IP address>)`: evaluates to `true` when the machine does not respond. It can be used when an attack is intended to crash the target machine.

Depending on the results of the active checking for a given attack, we can consider two cases:

- all the checks are positive, i.e. the attack was successful. An alert will be reported.
- some checks are negative, i.e. it was just an attempt and the attack failed. An alert with the appropriate `Alert.impact` field will be reported.

If, for a given attack, the proportion of failed (but detected) attacks over the successful ones becomes significant (and the target is found vulnerable to the attack), we would then suspect that the signature is leading to many false positives. The signature should then be reconsidered and improved.

3.2.3 The <REPORT> section. When the attack described in ADeLe is detected, an alert has to be sent. For that, the appropriate fields of an alert in the IDMEF format have to be filled. There are three kinds of values given to the fields of the alert: It can be:

- constant values (cf lines 103-104)
- values generated at runtime (cf lines 99,101)
- values coming from the events/alerts which led to the detection (cf lines 107,109)

3.3. THE RESPONSE PART

To our knowledge, there is no publicly known language designed to express automatic response in reaction to the detection of an attack. Automatic response can be a dangerous feature and sometimes the cure is worse than the disease. Indeed, as IP spoofing is common, the IP address of the supposed source of the attack may differ from the actual source of the attack. In this case, automatic reaction can lead to denial of services (DoS) because legitimate users can be denied access to data, services, or machines. However, we think that it would be useful to allow (even for a small number of attacks) expression of automatic response. A few precautions can be taken to limit the risk of DOS. For instance,

restrictions could apply only to external ¹¹ IP addresses or to normal users, rather than privileged users. We propose a (non-exhaustive) list of functions to be used for automatic response.

A first method to prevent further attacks is to deny temporarily access to a resource:

Close_Port(target_ip, "TCP" | "UDP" ,port_number) closes a port used for an attack^{1 2}.

Black_List (source_ip) adds an IP address to a black list^{1 3}.

In order to stop an attack in progress, more protective measures can be taken:

Reset_TCP_connection(source_ip,source_port,target_ip, target_port) terminates an offending TCP communication.

Kill_Process(target_ip,user_name,process_id|"ALL") kills one process (or all processes) from a particular user.

Shutdown_Machine(target_ip) shutdowns a machine remotely to ensure data integrity.

A more general function executes any script in reaction to the attack:

Script_Exec(script_name).

Such functions can be used in an ADeLe description in the <RESPONSE> part. Their parameters will then be either constants or values extracted from events/alerts which led to the detection.

4. CONCLUSIONS & FUTURE WORK

We presented in this article an attack description language. It has been designed to be easily readable. It can express both the attacker's and the defender's points of view. As the description of the attack contains (possibly executable) exploit code, a database of attack descriptions written in ADeLe makes it possible to test the efficiency of some IDS. The signature of the attack can be extracted from the description to allow configuration of probes and detection engines from a given IDS. Automatic response after the detection of the attack can be included in the description.

A preliminary version of a compiler from ADeLe to G^A_SSA_TA [11] exists and allows us to write signatures directly in ADeLe. In the near future, we plan to write compilers from ADeLe to other IDS.

¹¹ Outside the monitored network.

¹² Until the vulnerability has been corrected.

¹³ Used by a firewall.

Acknowledgments

This work was funded by the DGA/CELAR/CASSI as part of the Mirador project. The authors would like to thank all the members of the project for many useful comments and ideas.

References

- [1] M. Bishop. A standard audit trail format. Technical report, Department of Computer Science, University of California at Davis, 1995.
- [2] F. Cuppens and R. Ortalo. Lambda: A language to model a database for detection of attacks. In *Proceedings of the Third International Workshop on the Recent Advances in Intrusion Detection (RAID' 2000)*, October 2000.
- [3] D. Curry. Intrusion detection message exchange format, extensible markup language (xml) document type definition. `draft-ietf-idwg-idmef-xml-02.txt`, December 2000.
- [4] R. Deraison. The nessus attack scripting language reference guide. <http://www.nessus.org>, September 1999.
- [5] S. T. Eckmann, G. Vigna, and R. A. Kemmerer. Statl: An attack language for state-based intrusion detection. In *Proceedings of the ACM Workshop on Intrusion Detection*, November 2000.
- [6] R. Feiertag, C. Kahn, P. Porras, D. Schnackenberg, S. Staniford-Chen, and B. Tung. A common intrusion specification language (cisl). specification draft, <http://www.gidos.org>, June 1999.
- [7] J. D. Howard and T. A. Longstaff. A common language for computer security incidents. Technical Report SAND98-8667, Sandia National Laboratories, October 1998.
- [8] V. Jacobson, C. Leres, and S. McCanne. Tcpcdump 3.5 documentation. <http://www.tcpcdump.org>, 2000.
- [9] K. Kendall. A database of computer attacks for the evaluation of intrusion detection systems. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1999.
- [10] S. Kumar and E. H. Spafford. A software architecture to support misuse intrusion detection. Technical Report CSD-TR-95-009, The COAST Project Department of Computer Sciences, Purdue University, 1995.
- [11] L. Mé. Gassata, a genetic algorithm as an alternative tool for security audit trails analysis. In *Proceedings of the first international workshop on the Recent Advances in Intrusion Detection (RAID'98)*, 1998.
- [12] V. Paxson. Bro: A system for detecting network intruders in real-time. In *Proceedings of the 7th Usenix Security Symposium*, January 1998.

- [13] M. Roesch. Snort - lightweight intrusion detection for networks. In *Proceedings of the USENIX LISA '99 conference*, November 1999.
- [14] Secure Networks. *Custom Attack Simulation Language (CASL)*, January 1998.
- [15] Sun Microsystems, Inc. Sunshield basic security module guide. Solaris Documentation.
- [16] E. Turner and R. Zachary. Securenet pro software's snp-1 scripting system. White paper, <http://www.intrusion.com>, July 2000.
- [17] G. Vigna, S. T. Eckmann, and R. A. Kemmerer. Attack languages. In *Proceedings of the IEEE Information Survivability Workshop*, October 2000.

Appendix: An Example of Attack Description in ADeLe

The vulnerability exploited by the NFS_Mount attack is a bad configuration of the access rights of partitions remotely mounted via the NFS protocol. The attacker has initially a remote access level [9]. Thus the attacker can use several commands (`rpcinfo`, `showmount`, `finger`) to gather information about the potential target. He tries to find a home directory which is exported by the attacked system. If he finds such a home directory, he creates (on its local machine) an account with the same login name as the user owner of the remote exported home directory. Then, he mounts the remote partition via NFS and modifies/creates a “.rhosts” file containing the string “++”, thus allowing access to anybody from anywhere under this login. Finally, he can initiate a connection to the remote machine with the “`rlogin`” command: he has obtained a user access level [9].

This attack is described in ADeLe as follows:

```

1  Alert NFS_Mount (IN IPaddr targetip, OUT String account, OUT Connection cnx) {
2  <EXPLOIT>
3  <PRECOND>
4  Accesslevel == "REMOTE" #initial access level required
5  </PRECOND>
6
7  <ATTACK> <LANG> "EDL" </LANG>
8  String output; #gets everything displayed in the console
9  Integer ret_val; #exit code for the command
10 String rpc_services;
11 Integer ret_val0;
12 Connection shellhandler;
13
14 EVENT E0{
15 #Exec_shell_cmd(<shell_command>,<console_output>,<return_value>)
16 Exec_shell_cmd("rpcinfo -p "+targetip, rpc_services, ret_val0);
17 }
18 IF (ret_val0==0)&&("portmapper" IN rpc_services)&&("mountd" IN rpc_services){
19 Non_ordered{ #unspecified order!
20 [ Integer ret_val1;
21 String exported_partitions;
22 EVENT E1{
23 Exec_shell_cmd("showmount -e "+targetip, exported_partitions, ret_val1)
24 }
25 ]
26 [ Integer ret_val2;
27 String users_list;
28 EVENT E2{
29 Exec_shell_cmd("finger@"+targetip, users_list, ret_val2);
30 }
31 ]
32 }#Non_ordered
33 IF (ret_val1==0)&&(ret_val2==0){
34 String partition_found;
35 String user;
36 #Exists_exported_everyone(<input>,<partition_list>)

```

```

33     IF Exists_exported_everyone(exported_partitions,partition_found)
34         #Cross_part_users(<partition_list>,<users_list>,<matching_user>)
35     && Cross_partition_users(partition_found,users_list,user){
36         IF !Exists_local_user(user){
37             Add_local_user(user); #no observable event!
38         }
39     EVENT E3{
40         Exec_shell_cmd("mount -t nfs "+targetip+":/home/"+user+" /home/"+user,
41             output, ret_val)}
42     One_among{ #addition of "+ +" to the .rhosts file
43         [EVENT E4{
44             Exec_ shell _cmd("echo '+ +' >>~ ' ' +user+"/.rhosts",output,ret_val);
45         }
46         ]
47         [EVENT E5{
48             Exec_ shell_ cmd("echo '+ +' >~ " +user+" /.rhosts", output,ret_val);
49         }
50     ] # One _A mong
51     EVENT E6{
52         shellhandler: =Exec_cmd_shell("rlogin "+targetip+" -l "+account,
53             output,ret_val);
54         #now we have "User" access level
55         account:=user;
56         CNX :=shellhandler;
57     }
58 }
59 }
60 </ATTACK>

61 <POSTCOND>
62     Accesslevel := "USER"
63 </POSTCOND>
64 </EXPLOIT>

65 <DETECTION>
66 <DETECT>
67 <EVENTS>
68     #list of events types occuring during this attack (IDMEF notation)
69     E0 : Network.Classification[0].name == "rpcinfo -p"
70     E1 : Network.Classification[0].name == "showmount -e"
71     E2 : Network.Classification[0].name == "finger o''
72     E3 : Network.Classification[0].name == "mount home_directory"
73     E4 : System.Classification[0].name == "file append"
74     E5 : System.Classification[0].name == "file create"
75     E6 : Network.Classification[0].name == "rlogin"
76 </EVENTS>

77 <ENCHAIN>
78     E0 ; Non_ordered{E1 E2} ; E3 ; One_among{E4 E5} ; E6
79 </ENCHAIN>

80 <CONTEXT>
81     E4.File_Modified.name == ".rhosts"
82     E5.File_Created.name == ".rhosts"

```

```
83     IPAddr X := E0.Target[0].Node.Address.address
84     E1.Target[0].Node.Address.address == X
85     E2.Target[0].Node.Address.address == X
86     E3.Target[0].Node.Address.address == X
87     E4.Target[0].Node.Address.address == X
88     E5.Target[0].Node.Address.address == X
89     E6.Target[0].Node.Address.address == X
90     </CONTEXT>
91 </DETECT>

92 <CONFIRM>
93     #File_Contains(<file>,<contents>)
94     File_Contains("/home/"+E6.Target[0].User.name+"/.rhosts" , "+ +');
95 </CONFIRM>

96 <REPORT>
97     #construction of the returned alert here
98     Alert.version           := "1"
99     Alert.alertid          := NewAlertid()
100    Alert.impact            := "19"
101    Alert.Time.time         := NewTime()
102    Alert.Analyzer.ident    := "121212"
103    Alert.Classification[0].origin := "AdeLe'"
104    Alert.Classification[0].name   := "NFS_Mount"
105    Alert.Classification[0].url    := ""
106    Alert.Target[0].Node.Address.category := "2"
107    Alert.Target[0].Node.Address.address := E0.Target[0].Node.Address.address
108    Alert.Source[0].Node.Address.category := "2"
109    Alert.Source[0].Node.Address.address := E6.Source[0].Node.Address.address
110 </REPORT>
111 </DETECTION>

112 <RESPONSE>
113 </RESPONSE>
114 }
```

Sleepy Watermark Tracing: An Active Network-Based Intrusion Response Framework*

Xinyuan Wang[†], Douglas S. Reeves^{†‡}, S. Felix Wu^{††}, Jim Yuill[†]

[†] *Department of Computer Science*

[‡] *Department of Electrical and Computer Engineering*

North Carolina State University

USA

^{††} *Department of Computer Science*

University of California at Davis

USA

Key words: Intrusion Response, Intrusion Tracing, Active Security, and Network Security

Abstract: Network-based intrusion has become a serious threat to today's highly networked information systems, yet the overwhelming majority of current network security mechanisms are "passive" in response to network-based attacks. In particular, tracing and detection of the source of network-based intrusion has been left largely untouched in existing intrusion detection mechanisms. The fact that intruders can log in through a series of hosts before attacking the final target makes it extremely difficult to trace back the real source of network-based intrusions.

In this paper, we apply active networking principles to address the problem of tracing network-based intrusion with such chained connections, and propose a novel intrusion response framework: Sleepy Watermark Tracing (SWT). SWT is "sleepy" in that it does not introduce overhead when no intrusion is detected. Yet it is "active" in that when an intrusion is detected, the target will inject a watermark into the backward connection of the intrusion, and wake up and collaborate with intermediate routers along the intrusion path. By integrating a sleepy intrusion response scheme, a watermark correlation technique and an active tracing protocol, SWT provides a highly efficient and accurate source tracing on interactive intrusions through chained telnet or rlogin. Our

* This work has been supported by the Defense Advanced Projects Agency, administered by AFOSR under contract F30602-99-1-0540

prototype shows that SWT can trace back to the farthest trustworthy security gateway to the origin of intrusion, within one keystroke by the intruder. With its unique active tracing, SWT can even trace when intrusion connections are idle.

1. INTRODUCTION

Network-based attacks have become a major concern to today's highly networked mission critical information system. Existing network security mechanisms such as IDS, Firewall and IPSEC have not completely addressed the problem of network-based attacks. They are "passive" in front of network-based attacks and tend to be host-based. There is no automatic network-wide response even when attacks are detected.

One major problem in building an effective response to network-based attacks is the lack of source identification. Without effective source tracing, the attacked victim is blind at defending network-based attacks, and no effective intrusion countermeasures such as blocking and containing can be implemented. Network-based attacks can not be effectively repelled or eliminated until its source is known.

A complete solution to the problem of tracing network-based attacks is complicated by different anonymity gaining techniques used by different network-based attacks. For example, distributed denial-of-service (DDoS) attacks are usually generated from multiple previously-compromised slave machines, under control of a remote master machine. The unidirectional flooding traffic from slave machines usually comes with a "spoofed" source IP address, which makes it difficult to trace even the slave machines. For bi-directional, interactive intrusions, one of the most widely used techniques to conceal their true origin is to connect through "stepping stones"^[11] : intruders connect through a series of intermediate hosts before attacking the final target. All these techniques are easy to implement and use, making source tracing of network-based attacks among the hardest network security problems

In this paper, we focus on the real-time tracing of interactive intrusions that utilizes connection chains to disguise their source. A real-time solution to this problem not only enables us to stop or deter network-based intrusion near its source, but also helps to deter DDoS by better protecting hosts from being compromised into slave machines. While there are several approaches have been proposed to address the tracing problem of intrusion connection chains, they are all passive and tend to be isolated, and their lack of real-time network-wide coordination severely limits their practical use in real-time tracing in the current Internet.

On the other hand, active network^[2, 9] is an emerging framework that seeks to increase the programmability of computer networks and network components. It enables user and application to dynamically control how packets are handled. This customized packet processing opens new ways of securing networks that was not available in traditional passive networks.

In this paper, we apply active networks principle to address the problem of tracing network-based intrusion with chained connections, and present a novel intrusion response framework: *Sleepy Watermark Tracing (SWT)*. SWT is “sleepy” in that it does not introduce any overhead when there is no intrusion detected. Yet it is “active” in that when there is intrusion detected, it will trigger and coordinate network-wide tracing at real-time. SWT exploits the observations: 1) interactive intrusions with chained connections are bi-directional and symmetric at the granularity of connections; 2) application level contents are invariant across connection chains. By “injecting” carefully designed watermarks into the backwards-response traffic of the intrusion connection chain, SWT is able to trace through the intrusion connection chains at real-time - within a single keystroke by the intruder. Through its unique active tracing, SWT can trace through the connection chain even when the intruder is silent. All these represent substantial improvements over existing capabilities for tracing interactive intrusions with a chained connection.

In the next section, we discuss the general tracing problem and give a brief overview of existing tracing approaches. In section 3, we describe the general Sleepy Watermark Tracing method. In section 4, we present the SWT architecture, In section 5, we describe our prototype implementation of SWT and experimental results. In section 6, we conclude with possible future work.

2. TRACING PROBLEM AND APPROACHES

Given a series of computer hosts H_1, H_2, \dots, H_n ($n > 2$), when a person (or a program) sequentially connects from H_i into H_{i+1} ($i=1,2,\dots,n-1$), we refer to the sequence of connections on $\langle H_1, H_2, \dots, H_n \rangle$ as a *connection chain*, or *chained connection*. The *tracing problem* of a connection chain is, given H_n of a connection chain, to identify H_{n-1}, \dots, H_1 .

Tracing the source of intrusion through a connection chain over the Internet is a difficult problem. It requires network-wide collaboration among hosts in the network, and yet some of the hosts may be compromised and not trustworthy, As a network security mechanism, intrusion source tracing should be based on trust of appropriate network resources, and be robust against compromised hosts in the network. To trace back the chained

connections through multiple hosts, effective correlation is needed at intermediate nodes. Because network-based intrusion in today's high-speed network can be very short, correlation at intermediate nodes needs to be fast and accurate. Additionally, to scale the tracing system to the Internet, the tracing system should have minimum overhead while providing a fast response to detected network-based intrusion.

In general, tracing approaches for a connection chain can be divided into two categories: host-based and network-based, each of which can further be classified into either active or passive. Table 1 provides a classification of existing tracing approaches, as well as our proposed tracing mechanism.

Table 1. Classification of Existing Tracing Approaches and SWT

	Passive	Active
Host-based	DIDS CIS	Caller ID
Network-based	Thumbprinting Timing-based Deviation-based	IDIP SWT

Distributed Intrusion Detection System (DIDS)^[7] developed at UC Davis is a host-based tracing mechanism that attempts to keep track of all the users in the network and account for all activities to network-wide intrusion detection systems. Each monitored host in the DIDS domain collects audit trails and sends audit abstracts to a centralized DIDS director for analysis. While DIDS is capable of keeping track of all users moving around the network through normal login within the DIDS domain, it seems not feasible in large-scale network such as the Internet, because of its centralized monitoring of network activities.

The Caller Identification System (CIS)^[5] is another host-based tracing mechanism. It eliminates centralized control by utilizing a truly distributed model. Each host along the login chain keeps a record about its view of the login chain so far. When the user from the n -Ith host attempts to login into the n th host, the n th host asks the n -Ith host about its view of the login chain of that user, which should be 1,2 ... n -1 ideally. The n th host then queries host n -1 to 1 about their views of the login chain and so on. Only when the login chain information from all queried hosts matches, will the login be granted at the n th host. While CIS attempts to maintain the integrity of login chain by reviewing information from hosts along the login chain, it introduces excessive overhead to the normal login process.

Caller ID, described by Stuart Staniford-Chen^[3], is yet another interesting host-based approach that is said to be used by the Air Force. Caller ID is controversial in that it actually utilizes the same break-in technique used by intruders to break into the hosts along the connection

chain reversibly. If the intruder from H_0 connects through $H_1, H_2 \dots H_{n-1}$ to the final target H_n , the network security personnel at H_n first breaks into H_{n-1} ; from there they can find out the intruder comes from H_{n-2} , then they break into H_{n-2} and so on. Eventually they can find the origin of the intruder. One compelling advantage of Caller ID is that it is scalable to the Internet. It is also efficient in the sense that it introduces less overhead compared to DIDS and CIS. But its manual approach makes it difficult to trace short intrusion in today's high-speed network. Besides its legal complications, Caller ID also has the drawback that one must perform manual tracing on the host where the intruder is active, which is easily-noticed by the intruder.

The fundamental problem with the host-based tracing approach is its trust model. Host-based tracing places its trust upon the monitored hosts themselves. In specific, it depends on the correlation of connections at every host in the connection chain. If one host is compromised and is providing misleading correlation information, the whole tracing system is fooled. Because host-based tracing requires participation and trust of every host involved in the network-based intrusion, it is very difficult to be applied in the context of the public Internet.

Network-based tracing is the other category of tracing approaches. Neither does it require the participation of monitored hosts, nor does it place its trust on the monitored hosts. It is based on the property of network connections: the application level content of chained connections is invariant across the connection chain. In particular, the thumbprint^[3] is a pioneering correlation technique that utilizes a small quantity of information to summarize connections. Ideally it can uniquely distinguish a connection from unrelated connections and correlate those related connections in the same connection chain. While thumbprinting can be useful even when only part of the Internet implements it, it depends on clock synchronization to match thumbprints of corresponding intervals of connections. It also is vulnerable to retransmission variation. This severely limits its usefulness in real-time tracing.

The timing-based scheme^[11] by Zhang and Paxson is a novel network-based correlation scheme for detecting stepping stones across the connection chain. The correlation is based on the distinctive timing characteristics of interactive traffic, rather than connection contents. It pioneered new ways of correlating encrypted connections. It requires no clock synchronization and it is robust against retransmission variation. However, because its timing characteristics are defined over the entire duration of each connection to be correlated, it is difficult to be used in real-time correlation.

The deviation-based approach^[10] by Yoda and Etoh is another network-based correlation scheme. It defines the minimum average delay gap between the packet streams of two TCP connections as *deviation*. The

deviation considers both timing characteristics and the TCP sequence number, and it does not depend on the TCP payload. Similar to the timing-based approach, the deviation-based approach does not require clock synchronization and is robust against retransmission variations. However it is difficult to be used in real-time correlation as the deviation is defined over all the packets of a connection. Another drawback of deviation-based approach is that it correlates only TCP connections.

One fundamental problem with passive network-based approaches is its computational complexity. Because it passively monitors and compares network traffic, it needs to record all the concurrent incoming and outgoing connections even when there is no intrusion to trace. To correlate at any host in the connection chain, it needs to match every concurrent incoming connection with every concurrent outgoing connection at that host. That is, for a host with m concurrent incoming connections and n concurrent outgoing connections, the passive network-based correlation approach would take $O(m \times n)$ comparisons, in addition to the $O(m+n)$ scanning and recording of concurrent connections.

On the other hand, the active network-based approach dynamically controls how connections are correlated through customized packet processing. It does not need to record all the concurrent incoming and outgoing connections at any host in the connection chain. It does not need to match each concurrent incoming connection with each concurrent outgoing connection. For a host with m concurrent incoming connections and n concurrent outgoing connections, the active network-based approach is able to correlate within *time dependent only on the number of connections being actively traced*, in addition to the $O(m+n)$ scanning of concurrent connections.

IDIP (Intrusion Identification and Isolation Protocol) ^[6] is a proposal by Boeing's Dynamic Cooperating Boundary Controllers Program that uses an active approach to trace the incoming path and source of intrusion. In the proposal, boundary controllers collaboratively locate and block the intruder by exchanging intrusion detection information, namely, attack descriptions. While it does not require any boundary controller to record any connections for correlation, its intrusion tracing is closely coupled with intrusion detection. The effectiveness of IDIP depends on the effectiveness of intrusion identification through the attack description at each boundary controller. Therefore IDIP requires each boundary controller to have the same intrusion detection capability as the IDS at the intrusion target host. It is questionable whether the intermediate boundary controller is able to identify an intrusion based on a hard-coded attack description.

3. SLEEPY WATERMARK TRACING OVERVIEW

SWT is an active network-based tracing framework. It is "sleepy" in that it does not introduce overhead when no intrusion is detected. Yet it is "active" in that when an intrusion is detected, the target will inject a watermark into the backward connection of the intrusion and "wakes up" intermediate routers along the intrusion path.

By watermarking selected packets and processing them accordingly, SWT provides many potential advantages over existing intrusion tracing approaches. 1) SWT separate intrusion tracing from intrusion detection and it does not require any node other than the intrusion target to have the intrusion detection capability. 2) Unlike thumbprinting, timing-based and deviation-based approaches, SWT does not need to record all the concurrent incoming and outgoing connections at any node, and it does not require matching each of the incoming connections with each of the outgoing connections for correlation at any node. 3) SWT requires no clock synchronization and is robust against retransmission variation. 4) SWT traces only when needed. 5) So far the most compelling advantage of SWT is its correlation accuracy and efficiency. By using watermarks, SWT can trace the intrusion connection chain to its origin within a single keystroke of the intruder. With its unique active tracing, SWT can trace the intrusion connection chain back to its origin even when the intruder is inactive. 6) We have found that SWT can be implemented efficiently. It does not introduce any noticeable overhead to routers, and it only requires a few network server applications at the intrusion target host to be modified to inject watermarks.

In the remainder of this section, we describe the SWT model concepts and assumptions.

3.1 Basic SWT Concepts

In order to keep track of network-based intrusions to hosts, it is desirable to monitor hosts through the nearest router or gateways. This is termed a *Guardian Gateway*. We define the *Incoming Guardian Gateway* of host H as the nearest router that forwards incoming traffic to H and the *Outgoing Guardian Gateway* of host H as the nearest router that forwards outgoing traffic from H . It is possible that one host has more than one incoming or outgoing guardian gateway. We define the union of incoming and outgoing guardian gateways of a host as its *Guardian Gateway Set* (e.g., $\{GW_{in1}, GW_{in2}, GW_{out1}, GW_{out2}\}$ in Figure 1).

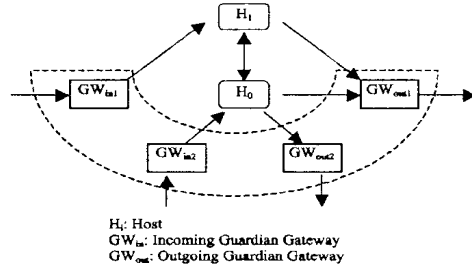


Figure 1: Guardian Gateway Set

For any guardian gateway set G , we define those hosts as *Guarded Host* of G whose guardian gateway set is a subset of G . For a host H , while the traffic between H and its directly-connected neighbour hosts does not pass through any gateways, the traffic between H and any non-directly-connected hosts must pass through its guardian gateway set.

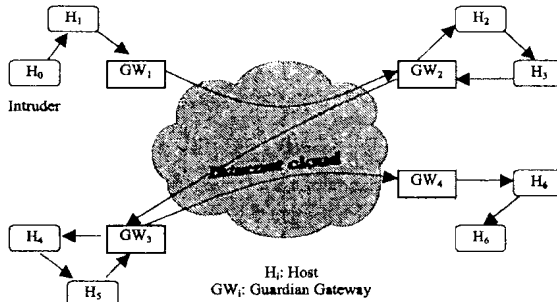


Figure 2: Tracing Model

We further define a *leap* as one connection step between hosts within a connection chain (e.g., $\langle H_i, H_{i+1} \rangle$ in Figure 2). One leap may consist of multiple hops (or links in the physical network) and the two guardian gateways of the two end hosts. A leap can be specified by a 5-tuple consisting of

\langle protocol number, source ip address, source port number, destination ip address, destination port number \rangle

Now the tracing problem of chained intrusion is defined as discovering and sequencing the guardian gateways of those hosts in the intrusion path, or (equivalently) as finding the leaps along the intrusion path.

3.2 Basic SWT Assumptions

We have identified the following assumptions that motivate and constrain our design:

- Intrusions are interactive and bidirectional,
- Routers are trust worthy and hosts are not trust worthy,
- Each host has a single SWT guardian gateway and
- There is no link-to-link encryption.

The first two assumptions represent our assessments of the nature of the intrusions. Here we refer to intrusions as those attacks aiming to gain unauthorized access, rather than denial of service attacks. A study of CERT security incidents ^[4] indicates that almost all security incidents, especially unauthorized access incidents, happened at computer hosts rather than routers or gateways. Therefore we believe our assumption to trust routers will cover most intrusion cases. In case there are indeed compromised routers involved in intrusion, the compromised router will be effectively indistinguishable from an attacker. The compromised router needs to be addressed first, before the tracing of the intrusion can go any further. In this case SWT can still trace to the farthest trustworthy guardian gateway.

The assumption of each host having a single SWT guardian gateway is only for simplifying the presentation of the SWT architecture. In case some host has multiple SWT guardian gateways, the guardian gateway set will be used in SWT tracing.

The final assumption represents the inherent limitation of any tracing based on network content. We believe that correlation of encrypted connections in real-time is still an open problem.

4. SLEEPY WATERMARK TRACING ARCHITECTURE

In general, the Sleepy Watermark Tracing Architecture consists of two complementing parties, namely, the *SWT guarded host* and the *SWT guardian gateway*. The SWT guarded host is the host that supports and thus is protected by SWT. The SWT guardian gateway supports SWT. In our trust model, each SWT guarded host has a unique SWT guardian gateway, and it maintains a pointer to its SWT guardian gateway. Each SWT guardian gateway may guard one or more SWT guarded hosts and it maintains the list of its SWT guarded hosts.

IDS and watermark-enabled applications at a SWT guarded host are SWT supporting components. In particular, IDS refers to an application level interface to any Intrusion Detection System ; this is the ultimate initiator of

SWT tracing. It interacts with SWT subsystem within SWT guarded host and triggers active watermark tracing once it detects an intrusion. Watermark enabled applications are those network service applications (such as telnetd, rlogind) that have been modified to inject arbitrary watermarks upon request.

The core of Sleepy Watermark Tracing consists of three interacting components: Sleepy Intrusion Response (SIR), Watermark Correlation (WMC) and Active Tracing (AT). In particular, Sleepy Intrusion Response accepts tracing requests from IDS, coordinates active tracing and keeps track of tracing information of intrusions. Watermark Correlation correlates incoming and outgoing connections through watermarks. Active Tracing coordinates different parties in the network to collaboratively trace the incoming path and source of intrusions.

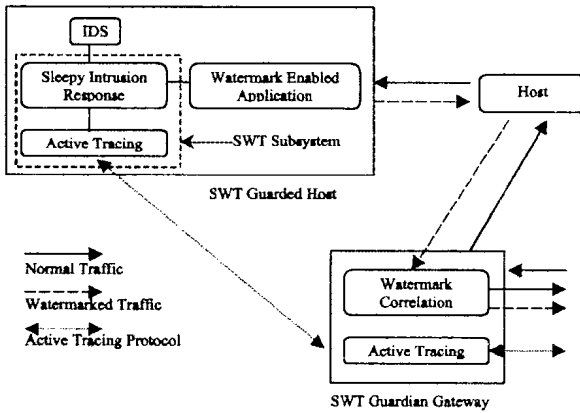


Figure 3: SWT Architecture

These three components work tightly together across SWT hosts and SWT guarded gateways. In specific, SIR and AT form the SWT subsystem within a SWT guarded host. Upon request from IDS, SIR coordinates a WM-enabled application and the AT module to initiate active tracing from the SWT guarded host to SWT guardian gateways. At the SWT gateway, the AT module receives tracing requests and provides watermarks to the WMC module. This module in turn provides AT module information about the next-leap SWT guardian gateway by correlating incoming and outgoing connections. Once the SWT guardian gateway finds next leap information about an intrusion connection chain, AT will send trace information to the original host that initiated the whole tracing and notify the next leap SWT guardian gateway to start watermark tracing.

4.1 Sleepy Intrusion Response

SIR controls and coordinates overall SWT intrusion tracing. It is in a SWT guarded host and it interacts with IDS and WM-enabled applications in the same host. To achieve high efficiency, SIR introduces “sleepiness” into SWT. By default, the SWT system is inactive and in sleep mode. When IDS detects an intrusion, it triggers SWT tracing by notifying SIR with appropriate connection information. Upon request from IDS, SIR first registers the intrusion connection as active for a configurable period of time, if it is not active already. Then SIR triggers active tracing on its guardian gateway by sending out trace notification; Finally SIR notifies the WM-enabled application that terminates the intrusion connection to start injecting the requested watermark. SIR also keeps track of tracing information of intrusions returned by the SWT guardian gateway, and upon request from IDS, SIR will provide tracing information on any specific active intrusion. If within a timeout period there is no trace information returned from the SWT guardian gateways, or further trace notification from IDS on an active intrusion connection, that intrusion response component will become inactive (“fall asleep”).

4.2 Watermark and Watermark-Enabled Applications

Conceptually, a *watermark* is a small piece of information that can be used to uniquely identify a connection. Ideally, a watermark should be easy to embed and retrieve and yet be invisible to normal users of network applications. In order to be used for correlation, a watermark must be able to traverse multiple connections and remain invariant (we assume that there is no encryption involved in the connections). Therefore, watermark belongs to the application layer and is application-specific.

One challenge in generating watermark is how to make watermarks invisible to end-users. For text based network applications such as telnet and rlogin, this is in many ways similar to hiding data in text ^[1], which is much more difficult than hiding data in pictures or sounds. The open space method is one of the major methods of data hiding in text files through manipulating white space. In particular, inserting spaces at the end of each line of text file will not be noticed by readers. But for network applications such as telnet and rlogin, simply inserting spaces will change the cursor position, and it is likely to be noticed by end users. Fortunately, the text being transferred to network applications is not necessarily the same as that being displayed. For example, the string

“**See me** *abc\b\b\b\b*”

transferred to telnet or rlogin will be displayed as the string

“See me”

We define a *virtual null string* of a network application as a string that appears null to end users of the network application. For instance, “*abc\b\b\b\b\b*” is a virtual null string of telnet and rlogin. Therefore by using virtual null strings, we can make watermarks invisible to such network applications.

In order to achieve high confidence of correlation, it is desirable to have the probability of collision of randomly generated watermarks as low as

$$P(m, n) = \prod_{i=1}^{n-1} \frac{m-i}{m} = \prod_{i=1}^{n-1} \left(1 - \frac{i}{m}\right)$$

possible. For $n > 1$ sites, assume each site independently generates a equiprobable random integer number between 1 and m , where $m \gg n$; let $P(m, n)$ be the probability such that those n random numbers are different from each other. Then we have:

When $m > n^2$, we have:

$$\prod_{i=1}^{n-1} \left(1 - \frac{i}{m}\right) > 1 - \frac{n^2}{2m}$$

Therefore, given $n = 2^{32}$, having $m \geq 2^{73}$ will make $P(m, n) > 0.999$. That means having 73 random bits in watermarks is sufficient to cover the whole IPv4 address space such that the probability of collision of generated watermarks is less than 0.1%.

Because the watermark is application specific, it needs to be injected into backward traffic through the application itself. *Watermark-enabled* applications are those network server applications (such as telnetd, rlogind) that have been modified to be able to “inject” requested watermark into their response traffic upon request. A watermark-enabled application processes two messages from SIR : WM-Start and WM-End, where WM-Start notifies watermark-enabled application to start injecting the enclosed watermark for specified times, and WM-End notifies the watermark-enabled application to stop injecting the watermark.

4.3 Watermark Correlation

In order to trace back along the intrusion connection chain, a mechanism is needed to find and match adjacent connections that belong to the same connection chain. We refer to this adjacent connection matching mechanism as *correlation*. According to the SWT tracing model, the hosts along the intrusion connection chain are not trustworthy, therefore, SWT is designed to correlate at SWT guardian gateways. Because the forward and backward

traffic of intrusion connection chain is symmetric at the granularity of leaps, watermarks along the backward traffic could be used for correlation at SWT guardian gateways.

By referencing its SWT guarded hosts, the through traffic of a SWT guardian gateway can be divided into two classes: *guarded* and *bypassing* (Figure 4). We define *guarded traffic* of a SWT guardian gateway as the traffic that either terminates at or originates from one of the SWT guardian gateway's guarded hosts, and *bypassing traffic* as all other traffic. It is obvious that the SWT guardian gateway needs to scan only the guarded traffic for possible correlation. We further define an *incoming leap* of a SWT guardian gateway as the connection that terminates at one of the gateway's guarded hosts, and an *outgoing leap* of a SWT guardian gateway as the connection that originates from one of the gateway's guarded hosts (Figure 4). Thus correlation at SWT guardian gateway can be modeled as matching an outgoing leap with an incoming leap.

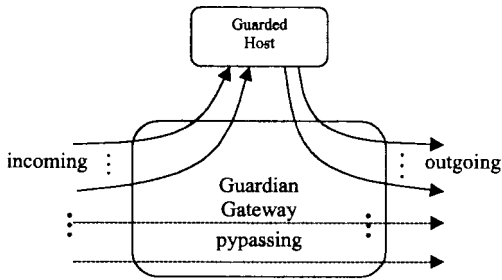


Figure 4: Guardian Gateway Correlation

One challenge of correlation at the SWT guardian gateway is that there may be multiple incoming and outgoing leaps through a single SWT guardian gateway. For a SWT guardian gateway with m incoming and n outgoing leaps, there are $m \times n$ combinations of possible matches after those $m+n$ leaps have been scanned. In specific, after m incoming leaps have been scanned, each of the n outgoing leaps scanned has m possible matches for correlation. Therefore exhaustive matching through multiple SWT guardian gateways would be complex and computationally expensive. To solve the connection matching combination explosion problem and achieve real-time response, SWT introduces the watermark as its basis for correlation.

With an identifying watermark injected to backward traffic of the intrusion connection chain, correlation at an intermediate SWT guardian gateways is simplified to scanning incoming and outgoing leaps and matching those with the same watermark. Specifically, when a SWT guardian gateway scans incoming leaps, it registers any leap that has a

registered watermark. When it scans outgoing leaps, it matches watermarked outgoing leaps with the incoming leap with same watermark.

The following observations can be made about watermark correlation:

- The accuracy of correlation is purely based on the uniqueness of the watermark, which is ultimately determined by SIR at the intrusion target host. This makes it possible to get very high confidence of correlation from tracing even a single watermarked packet.
- While the watermark is application specific, watermark correlation is generic. It has linear computation complexity across chained connections and requires no clock synchronization. Therefore it gives real-time response.

5. PROTOTYPE EXPERIMENTS

As a proof of concept, we have implemented a SWT prototype on FreeBSD 4.0. The prototype includes a SWT guarded host, SWT guardian gateways and a watermark-enabled application all running on the FreeBSD platforms.

We have performed two functional experiments on tracing a telnet connection chain: $A \Rightarrow B \Rightarrow C \Rightarrow D$, where A is the source of intrusion and D is the final intrusion target. The first is to trace the intrusion source while the intruder is active. Our SWT prototype demonstrates the capability of real-time tracing of a single watermarked packet: SIR at host D gets all the trace information back to intrusion source A within one key stroke from intruder at A. The second experiment is to trace the intrusion source while the intruder is inactive or silent. By actively sending back a watermark from watermark-enabled telnetd, our SWT prototype also gets all the trace information lead to the intrusion source A. As we have expected, for each watermarked packet, SWT triggers one GWTraceOn message travel from $D \Rightarrow C \Rightarrow B \Rightarrow A$, and two GWTraceInfo messages from C and B respectively.

To quantify the overheads incurred due to SWT itself, we have measured latency of SWT gateways with four different configurations:

- FreeBSD kernel IP forwarding without SWT;
- SWT configured to bypass traffic;
- divert socket IP forwarding without SWT;
- SWT configured to scan traffic.

The latency measurements were performed on a three node testbed configured in a straight line topology. The gateway at intermediate node was a 233Hz Pentium PC with 32 MB RAM, 512KB cache, and two Netgear FA310TX 10/100 fast Ethernet adapters, running FreeBSD 4.0.

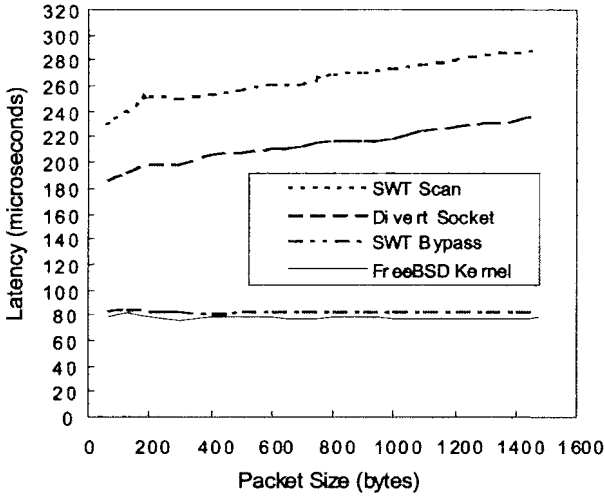


Figure 5: Latency of SWT Gateway

Figure 5 shows that the latency of FreeBSD kernel IP forwarding is about $78 \mu\text{s}$, independent of packet sizes. It takes about $83 \mu\text{s}$ for the SWT gateway to bypass and forward IP packets of various sizes. The $5 \mu\text{s}$ latency difference comes from IPFW rule matching in the FreeBSD kernel. The latency of divert socket IP forwarding ranges from $186 \mu\text{s}$ to $239 \mu\text{s}$ depending on the size of IP packets. The $103 \mu\text{s}$ to $156 \mu\text{s}$ overhead for divert socket forwarding over kernel forwarding includes: (1) overhead for two context switches for data reading and writing; (2) overhead for data copy in and out of user space; (3) overhead for dispatching system calls. Compared with divert socket IP forwarding, SWT scanning takes about $50 \mu\text{s}$ more time to forward IP packets of various sizes. This indicates that the SWT gateway latency overhead due to SWT itself is about $50 \mu\text{s}$.

6. CONCLUSIONS

In this paper, we have argued that network-wide, active intrusion response is needed in order to trace today's increasingly sophisticated network-based intrusions, which most likely utilize chained connections to hide their origin. We have presented SWT as an active network-based intrusion response framework and have shown that watermark can be used to construct highly accurate and efficient correlation for tracing chained intrusion connections. Our prototype shows that SWT is able to trace back to the trustworthy SWT guardian gateway that is closest to the source of intrusion chain, within single keystroke of the intruder. By actively injecting

watermark back to the intrusion connection, it is able to trace even when the intruder is silent.

By integration of Sleepy Intrusion Response, Watermark Correlation and Active Tracing, SWT provides highly effective, real-time and network-wide tracing of intrusions with chained connections. It is efficient, robust and scalable and it only requires some of the edge routers to participate tracing. Our experiment shows that SWT's own impact on a gateway's processing delay is only about 50 μ s.

These results lead us to conclude that active network technology can indeed provide better and yet practical solutions to some of the most difficult network security problems. It is our hope that SWT could be a building block for more active network security mechanisms such as dynamic perimeter defense, and dynamic intrusion blocking and containment.

REFERENCES

- 1 W. Bender, D. Gruhl, N. Morimoto and A. Lu. Technique for Data Hiding. *IBM Systems Journal*, Vol. 35, Nos. 3&4, 1996.
- 2 K. L. Calvert, S. Bhattacharjee and E. Zegura. Directions in Active Networks. *IEEE Communication Magazine*, 1998
- 3 S. Staniford-Chen, L. T. Heberlein. Holding Intruders Accountable on the Internet. In *Proceedings of IEEE Symposium on Security and Privacy*, 1995.
- 4 J. D. Howard. An Analysis of Security Incidents on The Internet 1989 - 1995, PhD Thesis, <http://www.cert.org/research/JHThesis/Start.html>, April 1997.
- 5 H. Jung, et al. Caller Identification System in the Internet Environment. In *Proceedings of 4th USENIX Security Symposium*, 1993.
- 6 D. Schnackenberg. Dynamic Cooperating Boundary Controllers. http://www.darpa.mil/ito/summaries97/E295_0.html, Boeing Defense and Space Group, March 1998.
- 7 S. Snapp, et al. DIDS (Distributed Intrusion Detection System) – Motivation, Architecture and Early Prototype. In *Proceedings of 14th National Computer Security Conference*, 1991.
- 8 X. Y. Wang. Survivability through Active Intrusion Response. In *Proceedings of 3rd IEEE Information Survivability Workshop (ISW-2000)*, October 2000.
- 9 D. Wetherall, J. Gutttag and D. Tennenhouse. ANTS: A Toolkit for Building and Dynamically Deploying Network Protocols. In *Proceedings of IEEE OPENARCH '1998*, April 1998.
- 10 K. Yoda and H. Etoh. Finding a Connection Chain for Tracing Intruders. In *F. Guppens, Y. Deswarte, D. Gollmann and M. Waidner, editors, 6th European Symposium on Research in Computer Security – ESORICS 2000 LNCS-1895*, Toulouse, France, October 2000.
- 11 Y. Zhang and V. Paxson. Detecting Stepping Stones. In *Proceedings of 9th USENIX Security Symposium*, 2000.

An Efficient Software Protection Scheme

Antonio Maña, Ernesto Pimentel

Computer Science Department

University of Málaga

29071 – Málaga SPAIN

e-mail: amg@lcc.uma.es

Key words: Software protection, smart cards, cryptography, information commerce.

Abstract: Software piracy has been considered one of the biggest problems of this industry since computers became popular. Solutions for this problem based in tamperproof hardware tokens have been introduced in the literature. All these solutions depend on two premises: (a) the physical security of the tamperproof device and (b) the difficulty to analyze and modify the software in order to bypass the check of the presence of the token. The experience demonstrates that the first premise is reasonable (and inevitable). The second one, however, is not realistic because the analysis of the executable code is always possible. Moreover, the techniques used to obstruct the analysis are not helpful to discourage an attacker with usual resources. This paper presents a robust software protection scheme based in the use of smart cards and cryptographic techniques. The security of this new scheme is only dependent on the first premise because code analysis and modification are not useful to break this scheme.

1. INTRODUCTION

Software protection is a complex problem; consequently there are several fields of research concerning different aspects of the problem. Some of the most important goals related to software protection are:

- *Intellectual property protection.* The objective is to link the software with information about its author. Among the

techniques used for this purpose the most popular is watermarking [CoTh99].

- *Protection against function analysis in mobile environments.* The objective in this case is to prevent a malicious host from discovering the purpose of a software agent and modify its behavior. Techniques like code obfuscation or function hiding [LoMo99] are used, sometimes complemented by the use of hardware tokens [Fünf99].
- *Protection against illegal copy and use of software.* The objective is to guarantee that only authorized users can run the software. Our work is mainly aimed to solve this problem.

Every year software industry has to face a cost of several billion dollars due to software piracy. In 1999, the global piracy rate for PC business software applications was 36 percent with an estimate cost of \$12 billion. As soon as computers started to become popular unauthorized copying of software started to be considered an important problem [Kent80]. Development of computer communications brought the growth of BBS services distributing pirated software. Today, other circumstances like the advances in code analysis tools and the popularity of Internet creates new opportunities to steal software. Some of the money lost because of the software piracy is included in the cost of legal software and therefore pirate copies are partially paid by the legal users.

Most of the software that is produced today has either weak protection mechanisms (serial numbers, user/password, etc.) or no protection mechanisms at all. This lack of protection is essentially derived from the user resistance to accept protection mechanisms that are inconvenient and inefficient. In Bruce Schneier words: “The problem with bad security is that it looks just like good security”. Many commercial software protection tools claim to achieve total security with software techniques. Most of these tools are *snake oil*¹. Theoretic approaches to the formalization of the problem have demonstrated that a solution that is exclusively based in software is unfeasible [Gold97].

On the other side, legal protection tools like *trade secrets*, *copyright*, *patents* and *trademarks*, are not adapted for the protection of software. Some

¹ Taken from the Snake-Oil FAQ: The term is used in many fields to denote something sold without consideration of its quality or its ability to fulfil its vendor's claims. This term originally applied to elixirs sold in travelling medicine shows. The salesmen would claim their elixir would cure just about any ailment that a potential customer could have. Listening to the claims made by some crypto vendors, “snake oil” is a surprisingly apt name.

authors have proposed the creation of new specific legal protection means for software products [Samu95].

An important related aspect is license management, that has to be capable of covering a wide range of situations and conditions while being easy and convenient for the final user.

Based on some advances of the general information security technology, we have developed a low cost software protection and license management scheme that is secure, flexible and convenient for the users. This scheme, avoids two of the most common attacks to software protection mechanisms: multiple installation from a single legal license and production of unprotected (pirated) copies of the software.

The rest of the paper is organized as follows. Section 2 reviews the most relevant related work. Section 3 introduces the new scheme. In section 4 we analyze implementation details. Other applications of this scheme are presented in section 5 and finally, section 6 summarizes the conclusions and presents ongoing research and future work.

2. RELATED WORK

In this section we will briefly review some proposals for software protection and license management, considering aspects like security, convenience and practical applicability.

One of the simplest and most popular protection mechanisms consists in a password or key check that enables installation of the software. If the check fails the software is not installed or it works in demo mode with restricted functionality. This mechanism is very popular in *shareware*. The password (or key) validation function is, evidently, included in the software. Therefore, it is possible to find it using reverse engineering. As a consequence it is frequent that key generation programs are produced by dishonest users and also that authentic passwords are published in certain Internet sites.

Sometimes the software is personalized to be used in one computer, for example, extracting information from some of the hardware devices (hard disk, network adapter, etc.) or from the operating system configuration. During its execution, the protected software checks that the computer is the one it was personalized for. This check, as the previous ones, can be bypassed. Also, this mechanism is inconvenient for the users because changes in the hardware or in the operating system may result in the need to get a new license and reinstall the software.

Self modifying code, and code obfuscation [CoTh00] are used in some software protection schemes. These techniques provide short term protection

and can be used in situations where software life is short (for example for agents and applets). Some of these techniques have been developed for a very special kind of software: virus [FHS97].

A very interesting approach is represented by function hiding techniques. In [SaTs98] the authors present a scheme that allows evaluation of encrypted functions. The idea is to establish an homomorphism between the space of cleartext data and the space of data enciphered by some cryptosystem. The objective is to evaluate some function on some data without revealing them. This process can be expressed this way: Let P be the domain of cleartext data and Q the domain of encrypted data. Let $f : P \rightarrow P$ be a function that the user wants to evaluate on some $x \in P$, and let $e : P \rightarrow Q$ and $d : Q \rightarrow P$ be respectively the encryption and decryption functions of some cryptosystem. Then, under certain conditions on the original function f , it is possible to find a function $f' : Q \rightarrow Q$ such that $\forall x \in P f(e(x)) = e(f(x))$ or, using an alternative of the previous expression $\forall x \in P d(f(e(x))) = f(x)$. This property is useful because it allows a piece of software to store $e(x)$ and implement f' in order to compute $f'(e(x))$ without revealing f , x or $f(x)$. Unfortunately this property only holds for certain families of functions (polynomial functions in this case).

Among the proposed solutions that rely on some hardware component, one of the most popular consists in the use of hardware tokens that are difficult to duplicate, which are connected through some communications port to the computer running the software. The protected software checks the presence of the token and refuses to run if the check fails. Examples of this kind of systems are hardware keys or dongles. These systems usually have the problem of the incompatibility between tokens of different applications. When the tokens are smart cards, as it is expected that the computer will include just one card reader, the user must continuously change the card, a problem known as *card juggling* that represents a serious inconvenience.

The check of the presence can be done in different ways; the simplest is to read a value from the communications port, but, commonly, to avoid that the interception of the communication in that port allows the attacker to replicate the token, the software will send a value (called challenge) that the token has to process, the software can predict the result that the token must send back. In any case, whatever the check is, it is not hard to bypass this protection, as the access to the communications port or the reader are easily found in the executable code. The check can then be bypassed obtaining a completely functional copy of the software as figure 1 shows. This process can even be automated by specially designed programs called "patches".

Sometimes the software is distributed encrypted and the token is used to decrypt it before it runs on the computer. The problem is that, when the software is decrypted, it is stored in the RAM memory of the user's (and

potential pirate) computer. There are different techniques that the pirate can use then to recover the software (for example producing a core dump).

One of the first proposals to use smart cards for software protection is presented in [ScPi84]. Protective technologies commercializes a tool that is based in those ideas and that share certain similarities with the initial scheme presented in the introduction of the section 3.

More recently, Aura and Gollman presented in [AuGo99] an interesting scheme based on smart cards and digital certificates that solves the card juggling problem and provides mechanisms for license management and transfer. In addition, a compilation of countermeasures against attacks are reviewed. Unfortunately, as their proposal is focused on the check of the presence of the smart card, it is vulnerable to the code modification attacks described above.

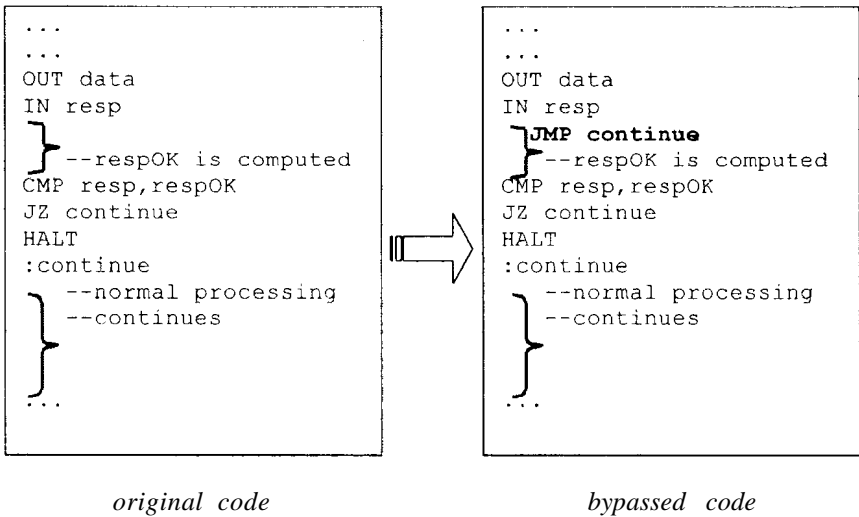


Fig. 1. Code modification to bypass the check of the presence of the token.

From the study of the problem it is concluded that to obtain a provable secure protection scheme we must have a tamperproof processor that contains and executes the protected software [HePi87]. A variation of this scheme is the distribution of encrypted code that the tamperproof processor decrypts and executes [Be94].

3. DESCRIPTION OF THE NEW SOFTWARE PROTECTION SCHEME

As it is usual in other fields of information security, in software protection there are no completely secure solutions. The objective of a software protection scheme is to make the attack to the scheme difficult enough to discourage dishonest users.

The new scheme is based, as others, in a tamperproof processor. The popularization of smart cards and their evolution in storage and processing capacity have lead us to consider them the most appropriate choice for our scheme. However, our design does not depend on this technology and, consequently, our solution can be implemented using any similar hardware token (for example, some hardware keys and some tokens that integrate smart card and reader functionalities).

A secure software protection scheme can be designed using just smart card technology. In this scheme some sections of the software to be protected can be substituted by functionally equivalent sections to be processed in the smart card. In this way, the protected software is divided and will not work unless it cooperates with the right card. Code modification attacks will not succeed in this case. In fact, the only possible attack is to analyze the data transmitted to and from the card trying to guess the functions that the card performs. If we include enough functions, with enough importance in the main code, and enough complexity, the attack described could become impractical.

This scheme needs one card per application and the quantity and complexity of the protected functions are limited by the capacity of the card. Moreover, this scheme does not allow the distribution of the protected software using Internet because the cards must be distributed with the software. With the purpose of avoiding the aforementioned problems we will introduce the cryptography as the second building block of our software protection scheme.

3.1 Fundamentals of the new scheme

Figure 2 shows the first scheme that we elaborated. We will use it to illustrate the final scheme. The figure shows that several sections of the original code are substituted by their equivalent for the card during the production phase. These new sections are encrypted with the public key of the card using an asymmetric cryptosystem [RSA78] during the personalization phase and are kept encrypted so only the card that has the matching private key will be able to decrypt and execute those protected sections. The cards now have to store a key pair, but the protected software

sections do not reside on the cards. The key pair must be generated in the card and the private key must never be transmitted outside the card. The original code sections are substituted by calls to a function that transmits their equivalent protected sections (e.g. “B”), including code and data, to the card, where they are decrypted and executed. When finished, the card sends back the results.

Assuming that the encryption algorithm is secure, the attack to the system must be based in the analysis of the input and output data (and possibly the running time) of the card functions. However, we must emphasize that now the card only stores one function at a time and therefore we can use more complex functions because all the capacity of the card is now available for each single function. Moreover, this scheme allows the card to execute any number of protected functions. The dishonest user will need to discover all of the protected functions to be able to break this protection scheme.

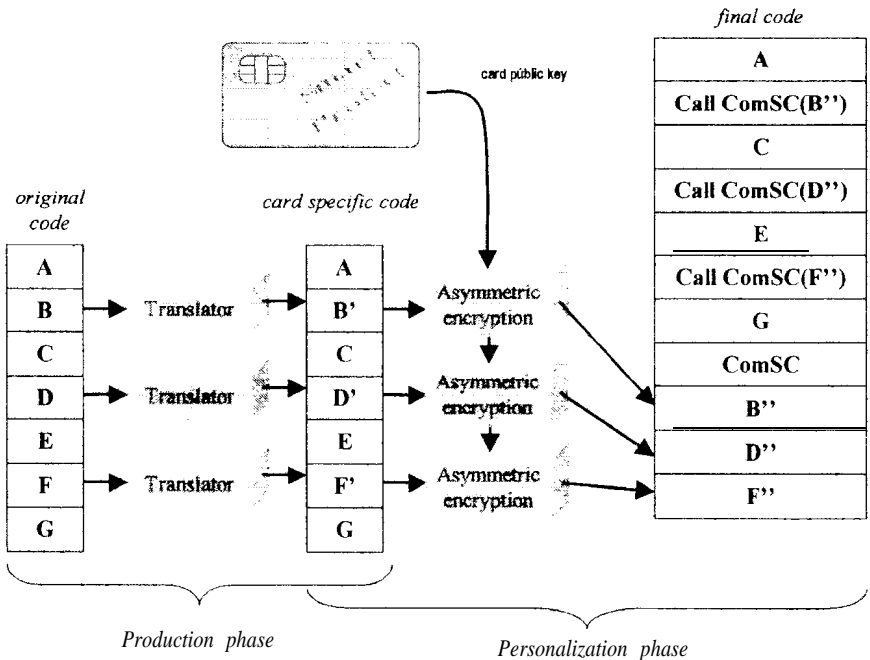


Fig. 2. Code transform in our first software protection scheme.

An alternative attack could consist in the substitution of some of the authentic protected sections by other fake sections produced by the dishonest user (for example such a false section could try to send back the contents of

the card). This attack can be considered a kind of “Trojan horse”. To avoid these attacks we must authenticate the code before its execution [DDB89].

To summarize, this first scheme allows a single card to be used to protect many applications, increases the complexity of the protected functions, allows the card to execute any number of those functions and enables the distribution of the software through Internet.

But, in spite of the advantages mentioned, some aspects like efficiency and robustness of the scheme need to be improved. The use of an asymmetric cryptosystem introduces a high computational cost. Also the lack of a code authentication mechanism opens a dangerous attack line. On the other hand, this first scheme does not take into account some desirable features like license transfer or expressive authorization. Also, the need to include a personalization phase is not adequate for some distribution models. We want the software to be freely distributed, although to run it the user will need to get a license.

The final scheme is shown in figure 3. In this case the production phase includes the encryption of the protected sections (wich include code and data) with a symmetric cryptosystem.

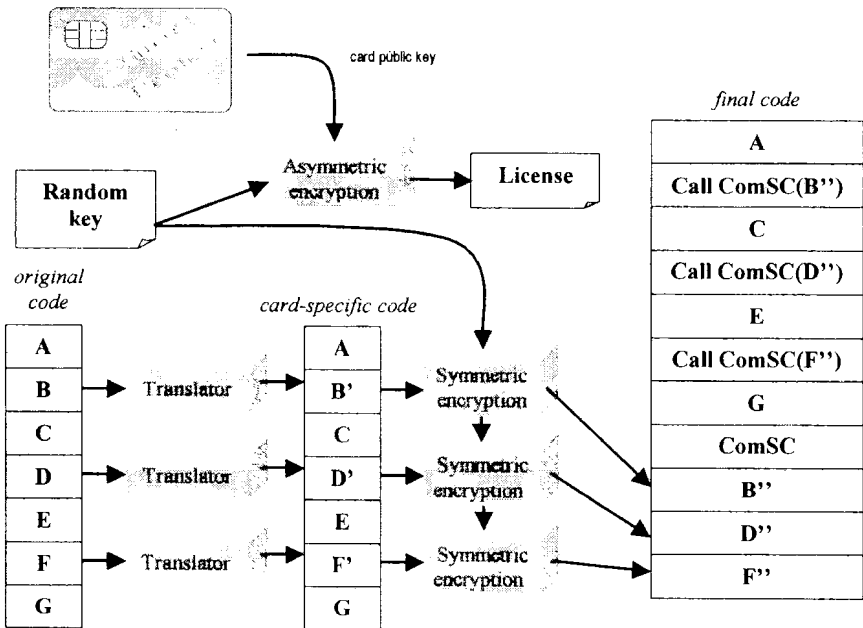


Fig. 3. Code transform and license production.

In the authorization phase (equivalent to the personalization phase of the previous scheme), a new license is produced containing the random symmetric key used to encrypt the protected sections, information about conditions of use (i.e. time limits, number of executions, etc.), the identification of the software (ID, version number, etc.) and finally the identification of the license. All this information is encrypted with the card public key. When the license is received by the client it is stored in the card.

The functionality of the previous scheme is maintained in this new one, but the efficiency is improved because decryption of the protected sections is now much faster. The definition of the license structure permits a high degree of flexibility. Furthermore, as each application has its own key, we can manage them individually.

We previously mentioned the necessity to authenticate the code to be executed by the card to avoid certain attacks. In this scheme, because the protected sections are encrypted using a symmetric key that is kept inside the card, it is impossible for a dishonest user to produce false sections. However, if the license was to be transmitted using an insecure channel, a man-in-the-middle attack could be carried out, but as we will show in the next section, the software producer will require a certificate of the card public key that the dishonest user will not be able to forge.

3.2 License management

3.2.1 Sale

Because the license for the user (containing the key to decrypt the protected sections) is encrypted with the card public key, it is essential to avoid that the corresponding private key is known outside the card. To achieve this objective the most practical solution is to use special smart cards produced for this purpose. These cards will contain a key pair and some support software. A certificate of the public key of the card is signed by the card manufacturer to guarantee the authenticity of the keys.

To buy a protected application, the client sends a request containing the certificate of the public key of his card and a random number to the software producer. The producer verifies the validity of the certificate and, in case the validation succeeds, produces a new license, encrypts the license and the random number using the public key received and sends it to the client card. The card verifies that the license matches the request (i.e. the random number is correct) and stores it. The producer also stores all the licenses in a

database to be able to generate new licenses for the client when needed (theft, destruction of the card, etc.). If a request for an already generated license is received, the producer will prepare a new license for the client with no extra cost. This new license will include a different serial number (this number is part of the identification of the license). The software application is distributed and copied freely, with no additional protection.

3.2.2 Transfer

One of the features that we have considered important (introduced in [AuGo99]) is license transfer. License transfer could be used to delegate the right to use some software application to another user or simply to store your license in a new card. Our scheme introduces the possibility of selective license transfer.

Our license transfer scheme has been designed to avoid using certificate chains because of the overhead in communication, storage and processing they introduce. Another goal was to avoid storing public keys of external entities in the smart cards.

The protocol to transfer a license is divided in two phases: *delegation* (steps 1 to 3) and *recover* (steps 4 to 6). We call this protocol *direct transfer* opposed to the *scheduled transfer* which is used mainly for recovery purposes. The protocol is as follows:

1. The user selects which license (or licenses) are going to be transferred from the source card. Notice that, opposite to other systems, our scheme does not oblige the user to transfer all the licenses in the source card (which we consider to be a serious limitation). In the rest of this protocol we will assume that we are transferring one specific license.
2. The public key certificate of the destination card is sent to the source card.
3. The source card creates a certificate delegating the license to the public key of the destination card, destroys its own license and finally sends the delegation certificate to the destination card.
4. The destination card requests a new license to the software producer. This request includes the delegation certificate received from the source card and the destination card public key certificate.
5. The software producer verifies both certificates and generates a new license for the destination card if the verification succeeds. The license database is updated accordingly.
6. The destination card decrypts and stores the new license.

Suppose now, that the protocol described above is interrupted (accidentally or intentionally to attack the protection scheme). For instance, if the protocol is aborted after step 3, the destination card would possess the delegation certificate but not the new license. The source card has already destroyed its license but it can request a new copy from the software producer and get a new valid license. Afterwards, using the delegation certificate that has stored, the destination card can also get a new license. This attack could be used to replicate any number of licenses.

To prevent this attack, a serial number, different for each new copy of the license produced, is included in the license (see section 3.2.1). In the scenario depicted above, when the source card requests the new copy after aborting the protocol, the software producer generates a new license (with a different serial number) that is sent to the card and stored in the database. Later, when the destination card attempts to use the delegation certificate to get a new license, the request will be denied.

If the protocol is aborted during the step 3 (for instance, extracting the card from the reader) it may occur that the source card have destroyed its license and the delegation certificate has not been sent to the destination card. In this situation the source card can still request a new license.

The inclusion of the software producer in the transfer protocol may seem inconvenient but if the producer is not included, the source card would need to verify the public key certificate of the destination card which, in turn, would increase the complexity of the protocol and also would introduce weaknesses in the protection scheme.

3.2.3 Recovery

Providing efficient and convenient solutions to the problems that the protection scheme may introduce is considered very important for user acceptance. In any scheme that uses some kind of hardware components it is essential to prevent the consequences of failure in those components. In our scheme licenses are linked to smart cards based on the fact that the private key is not known outside the card. Consequently, in case of card failure it will be impossible to run the software. For this contingency, the user must take some prevention measures. As the price of the cards is small, it seems reasonable to prepare a replacement card to be used in case of failure of the main card. The preventive process requires the execution of the delegation phase of the scheduled transfer protocol for all the licenses in the card. In case of failure of the main card, the protocol would continue on the recover phase. At the end of the protocol the replacement card will possess the same licenses as the main card.

The difference between the direct transfer protocol and the scheduled transfer protocol is the inclusion of the date (or other parameter like number of executions) when the transfer will take place. This date is included in the delegation certificate. Steps 3 and 4 of the direct transfer protocol are replaced by this sequence in the scheduled transfer protocol:

- 3'. The source card creates a certificate delegating the license to the public key of the destination card on date D and sends it to the destination card. The source card will not delegate that license again to any other card until date D .
- 4'. Later two different situations can arise:
 - If the user wants to keep using the main card the replacement card must destroy the delegation certificate and send a new scheduled transfer request before date D . In this case the source card will accept the request.
 - Otherwise, on date D :
 - Source card will destroy its own license.
 - As in the direct transfer case, both cards can request a new license to the software producer but only the first will be accepted.

3.2.4 License expiration

The licenses are always kept protected because they are either encrypted or stored in the smart card. Therefore, the card software, which is trustworthy, can destroy licenses when they expire (we can use different parameters like number of executions, time of use, etc.), the software can even warn the user when the expiration is about to happen. One of the parameters most used in software licenses is the expiration date. To include this parameter it would be interesting to have an internal clock in the cards. Some manufacturers have announced cards including this feature.

4. IMPLEMENTATION DETAILS

Today, smart card technology offers features that not so many years ago corresponded to personal computers [CDHP00]. However, compared to the processing power of the host computers, each access to the smart card introduces important delays. As our scheme requires the transmission of a considerable amount of code and data to and from the card, it is important to take into consideration the efficiency of the protection scheme.

The amount of data and code transmitted determines the magnitude of the delay introduced. On the other side, since the main attack to the protection

scheme is based in the analysis of the functions performed by the card, the protection scheme will be more secure as the functions grow in size and complexity.

Consequently, it is necessary to find a balance between security and speed. Fortunately, in this case, this balance is possible and it is not difficult to obtain security and speed measures that satisfy both the software producer and the client. A detailed description and study of the efficiency of the protection scheme is included in [LMP00].

The scheme has been designed and the tests carried out using smart cards with symmetric and asymmetric cryptographic capabilities. An implementation that uses smart cards that have only symmetric cryptographic capabilities is possible, but the changes that need to be introduced in the scheme, together with the low prices of the cards with both types of cryptosystems, do not justify the use of cheaper cards.

4.1 Functions executed by the smart cards

This is an essential characteristic because the security of the system is based on the difficulty to guess the functions that the card executes from the analysis of the input and output data and the execution time [Hohl98].

If we know that the function performed by the card represents a straight line then we just need to run the function two times with different input data to discover it. In contrast, functions like one-way hashes [Pren00] or digital signatures [RSA78] are not vulnerable to these attacks. In most software applications this type of functions is not used frequently, but the functions that appear in most software applications have an advantage: they have more input and output data.

To make it difficult for the pirate to analyze the functions we include false (dummy) input and output data that are not used for the computation of the function, although it is transformed to confuse the attacker. Another technique that is very effective to obstruct the analysis is to mix the processing of several functions with the intention that the result of each call to the card depends on the input data of the previous calls and even on results of previous calls that have not been send back as results but stored in the card memory.

4.2 Card readers

One of the most common kind of software piracy takes place inside the organization of a legal client of the software by the use of multiple copies of a legally acquired software application. In our scheme this attack could be carried out making several computers share a card reader.

This problem has been considered in previous schemes, but the most common solution is to make the software have direct access to the card reader. This solution introduces countless problems and computational costs in the protected software because it must manage different situations and hardware features that are usually managed by the operating system.

In our scheme, to prevent this attack we have designed a solution based on the last technique described in section 4.1. The system “chains” the calls to the card so any incorrect sequence of calls (produced if several computers share a card reader) will result in the software producing erroneous results.

5. OTHER APPLICATIONS

The scheme introduced can be useful in other environments, in fact it was devised from a previous work on information commerce over Internet [Mana00]. As an example of the different possibilities of this scheme we will explain briefly how it can be used for information commerce in applications like online newspapers [Const97] or digital libraries [KLK97].

For this application each user must possess a special smart card (with a key pair and our base software), a card reader and a web browser that can access the card (i.e. with a special *plug-in*).

To gain access to some information the client sends a request to the information provider, including the public key certificate of the client’s card. This step might implicate some negotiation of the conditions of the trade. The information provider, using the applet generator described in [Mana00] generates a specific applet to fulfill the request and a license for the client’s card. This applet includes protected sections that have to be executed by the card using the license. Because the card software is trustworthy we are able to control aspects like number of executions and, what is more, we can include an electronic purse to pay for the information accessed.

6. CONCLUSIONS AND FURTHER WORK

We have described a robust software protection scheme based in the use of smart cards and cryptographic techniques. Related schemes based in tamperproof hardware tokens that have been proposed in the literature have been analyzed concluding that all of them are based in the check of the presence of the token and are therefore vulnerable to code modification attacks. Considering that the new scheme is not based in that check, code modification is not a potential attack. We have shown the different protocols for the management of licenses and analyzed the security of the scheme and

the importance of the implementation details. Finally, we have also introduced possible alternative applications of the scheme. Hence, we can conclude that the advantages of the presented scheme are robustness against different attacks (bypassing the check, code substitution and attacks to the license management protocols), confidence for the user, efficient use of the computational resources of the smart cards, free distribution and copy of the software, selective license transfer, control of the expiration of the licenses and applicability in distributed computing environments.

Tools to produce protected software automatically from unprotected executable programs, applet protection and payment integration are under development. We are studying the possibilities that the combination of function hiding techniques with our scheme could open.

Finally we are studying the security achieved by the different families of functions that can be executed in the cards to obtain a measure of the protection achieved in some particular software application.

REFERENCES

- [AuGo99] Aura, T.; Gollman, D. *Software License Management with Smart Cards*. Proceedings of the Usenix Workshop on Smartcard Technology (Smartcard'99), pp. 75-86. 1999.
- [Be94] Bennet S. Yee. *Using Secure Coprocessors*. PhD thesis CMU-CS-94-149, Carnegie Mellon University, 1994.
- [CDHP00] Castellá-Roca, J.; Domingo-Ferrer, J.; Herrera-Joancomarti, J.; Planes, J. *A Performance Comparison of Java Cards for Micropayment Implementation*. Proceedings of CARDIS'2000, pp 19-38. Kluwer Academic Publishers. 2000.
- [Cons97] Constantas, D. et al. *An Architecture for Electronic Document Commerce*. 4th CaberNet Radicals Workshop, 1997. Available online at <http://www.newcastle.research.ec.org/cabernet/research/radicals/1997/papers/edc-constant.html>
- [CoTh00] Collberg, C.; Thomborson, C. *Watermarking, Tamper-Proofing, and Obfuscation - Tools for Software Protection*. University of Auckland Technical Report # 170. Available online at <http://www.cs.auckland.ac.nz/~collberg/Research/Publications/CollbergThomborson2000a/index.html>. 2000.
- [CoTh99] Collberg, C.; Thomborson, C. *Software watermarking: Models and dynamic embeddings*. Proceedings of POPL'99 - 26th ACM Symposium on Principles of Programming Languages. 1999. Available online at

<http://www.cs.arizona.edu/~collberg/Research/Publications/CollbergThomborson99a/index.html>. 1999.

- [DDB89] Davida, G. I.; Desmedt, Y.; Blaze, M. J. *Defending Systems Against Viruses Through Cryptographic Authentication*. Proceedings of IEEE 1989 Symposium on Security and Privacy, pp 312-318. 1989.
- [FHS97] Forrest, S.; Hofmeyr, S.; Somayaji, A. *Computer immunology*. Communications of the ACM, Vol. 40, No. 10, pp. 88-96. 1997.
- [Fünf99] Fünfroeken, S. *Protecting Mobile Web-Commerce Agents with Smartcards* Proceedings of ASA/MA'99. 1999.
- [Gold97] O. Goldreich, *Towards a theory of software protection*, Proc. 19th Ann. ACM Symp. on Theory of Computing, pp. 182-194. 1987.
- [HePi87] Herzberg, A.; Pinter, S. S. *Public Protection of Software*. ACM Transactions on Computer Systems, 5(4)-87, pp. 371-393. 1987.
- [Hohl98] Hohl F. *Time Limited Blackbox Security: Protecting Mobile Agents from Malicious Hosts*. in Giovanni Vigna (Ed.), *Mobile Agent Security*, LNCS 1420 Springer Verlag, pp 91-113. 1998.
- [Kent80] Kent, S. *Protecting Externally Supplied Software in Small Computers*. PhD thesis, Massachusetts Institute of Technology, MIT/LCS/TR-255, MIT. 1980.
- [KLK97] Kohl, U.; Lotspiech, J.; Kaplan M. A. *Safeguarding Digital library Contents and Users: Protecting Documents Rather Than Channels*. D-Lib Magazine, Sept-97 ISSN 1082-9873. 1997.
- [LoMo99] Loureiro, S.; Molva, R. *Function hiding based on error correcting codes*. Proceedings of Cypotec'99 - International Workshop on Cryptographic techniques and Electronic Commerce. 1999.
- [Mana00] Maña, A. *Una Solución Segura Basada en Java para la Comercialización de Contenidos Digitales*. (in spanish). Proceedings of the Sixth Spanish Conference on Cryptography and Information Security. Ra-Ma, isbn 84-7897-431-8, pp-243-252. 2000.
- [LMP00] López, J.; Maña, A; Pimentel, P. *Un Esquema Eficiente de Protección de Software Basado en Tarjetas Inteligentes*. Technical Report 14/2000, Department of Computer Science, University of Malaga. 2000
- [Pren00] Preneel, B. *El Estado de las Funciones Hash*. (in spanish). Proceedings of the Sixth Spanish Conference on Cryptography and Information Security. Ra-Ma, isbn 84-7897-431-8, pp-3-38. 2000.
- [RSA78] Rivest, R. L.; Shamir, A.; Adleman, L. M. *A method for obtaining digital signatures and public-key cryptosystems*. Journal of the ACM, 21(2):120-126, February 1978.
- [Samu95] Samuelson, P. *A Manifesto Concerning the Legal Protection of Computer Programs: Why Existing Laws Fail To Provide Adequate Protection*. Proceedings of KnowRight '95, pp 105-115. 1995.

- [SaTs98] Sander, T.; Tschudin C.F. *On Software Protection via Function Hiding*. Proceedings of Information Hiding '98. Springer-Verlag. LNCS 1525. pp 111-123. 1998.
- [ScPi84] Schaumüller-Bichl, I.; Piller, E. *A Method of Software Protection Based on the Use of Smart Cards and Cryptographic Techniques*. Proceedings of Eurocrypt'84. Springer-Verlag. LNCS 0209, pp. 446-454. 1984.

This Page Intentionally Left Blank

Protecting the Creation of Digital Signatures with Trusted Computing Platform Technology Against Attacks by Trojan Horse Programs

Adrian Spalka, Armin B. Cremers and Hanno Langweg

Department of Computer Science III, University of Bonn

Roemerstrasse 164, D-53117 Bonn, Germany

Email: adrian@cs.uni-bonn.de

Key words: Digital Signatures, Trojan Horse Programs, Trusted Computing Platform

Abstract: Digital signatures are a key technology for many Internet-based commercial and administrative applications and, therefore, an increasingly popular target of attacks. Due to their strong cryptographic properties an attacker is more likely to subvert them with malicious software, ie Trojan horse programs. We show that by fusing two techniques, our WORM-supported reliable input method and the Intelligent Adjunct model of the Trusted Computing Platform Alliance, we can achieve a high degree of protection from Trojan horse programs during the process of creating digital signatures. Existing software products immediately benefit from our results. Moreover, we examine three ways of storing and executing the signing software with respect to its susceptibility to Trojan horse programs and identify the most suitable combination.

1. INTRODUCTION

Digital signatures are recognised by the industry, the government and the administration, to name just a few, to be the driving technology for the provision of a wide spectrum of Internet-based services. The permanent success of business-to-consumer e-commerce, e-government, e-administration – one is tempted to say e-everything in view of the recent developments – depends, by and large, on the acceptance of these services

by a substantial number of ordinary private end-users. This will only come if the services are equipped with a sufficient level of security.

Confidentiality of a communication over the Internet is enforced by encryption algorithms. The digital signature ensures its integrity and the identification of the communicating parties or, to be more precise, the digital signature either confirms the integrity of a document and the claimed identity of its sender or uncovers a discrepancy or a lack of correspondence. By their definition, digital signatures employ cryptographic algorithms, usually a hash-function and a public-key system.

The cryptographic algorithms used today are strong in the sense that the computing power necessary to forge a digital signature or to break confidentiality is unavailable to any individual, party or even institution. Since this claim is backed by both theoretical and practical results, an adversary easily recognises that an attack on the cryptographic algorithms is of little avail. Though as such highly positive, for this setting to be invincible one assumes that the adversary has access only to the data travelling over the Internet and, in particular, has no access to the computers of the communicating parties.

While no user would deliberately start a malicious program on her PC, the history of attacks based on viruses, worms, Trojan horses and other rogue programs convincingly demonstrates that an attacker can have various ways to start an attack on the user's PC. The Internet is also a perfect means for distributing such programs. Trojan horse programs often come camouflaged as a nice utility, which offers the user some useful function and, at the same time, performs malicious functions in the background. The user just needs to down-load and install it. Whereas an organisation or a company can establish a security policy that minimises such threats, the end-user at home, ie, the envisaged group of digital signature holders, will consider down-loading and running programs, plug-ins etc. from the Internet a normal operation – or at least their children do so.

The threat posed by Trojan horse programs is real. In their comprehensive investigation of current issues related to electronic commerce Lacoste *et al* (2000)¹ state:

In spite of sufficiently secure existing algorithms, the technical possibility of obtaining signatures in an underhanded way, aided by such mechanisms as Trojan horse attacks, cannot be ignored. Such attacks might result in an unpredictably high damage for the key holder, especially if he cannot prove that an attack has happened. Hundreds or thousands of transactions can be made within a short time by an attack. [. . .] This means: A software solution for digital signatures might be

¹ Cf Lacoste *et al* (2000), pp 233.

completely correct and nevertheless cannot prevent malicious Trojan horse attacks.

This statement goes in line with our observation. The digital signature is expected to gain the same level of validity as the human manual signature does any time soon. Due to its universal usability, eg, it can be used for buying a car, casting a ballot in a vote or changing the registration of a car over the Internet, it will be a popular target for attacks. And since the cryptographic algorithms cannot be broken, Trojan horses are likely to subvert the process of digitally signing documents.

We present some of the current approaches of dealing with Trojan horses, or generally, with untrustworthy programs in the subsequent section. One can observe there two extreme views. It is either assumed that all programs on the PC are trustworthy or that any program can be untrustworthy. We believe that, for practical purposes, both are unrealistic. On the one hand, there can always be one untrustworthy program on a PC, on the other, some kinds of programs are more likely to be untrustworthy than others due to the effort on the attacker's side. To give an example, it is much more difficult and complex for an attacker to replace a system internal driver than to write a trendy utility.

On these grounds we have decided to examine the combination of two techniques for strengthening the defence against attacks by Trojan horses on digital signatures. Firstly, the Trusted Computing Platform Alliance, a group of almost 150 companies, including major industry players, promotes an Intelligent Adjunct model presented by Balacheff et al (2000). It aims to ensure that a computer's key components, including major parts of the operating system, are checked for their integrity before being used. This check does not guarantee the absence of Trojan horses, but, if passed, it guarantees that these components have not been modified in an unauthorised manner. Thus, if we can take that the official hardware and system software vendors do not implant malicious functions in their products, then the checked part of the PC constitutes a trusted software subset. Therefore, if all components involved in the creation and verification of a digital signature can use this trusted subset to check their own integrity, then some attacks by Trojan horses can be prevented and some detected. The group of remaining attacks becomes significantly smaller and significantly harder to assemble and to put in place innocuously. Thus, secondly, we propose the inclusion of an inexpensive software WORM medium, which ensures a reliable, ie, unmodified input to the signing software.

Moreover, we address three scenarios of storing and executing the signing software:

- signing software installed on the local hard disk and executed as a regular program
- storage on smart card, execution in secure JVM
- storage and execution on smart card, employing intelligent adjunct technology

We show that, together with our previous findings, the use of a JavaCard as secure software storage and the intelligent adjunct technology we gain higher security and platform-independence.

2. PREVIOUS AND RELATED WORKS

Currently there are three different approaches to facilitate the use of digital signatures in insecure environments. We refer to them as ‘secure hardware’, ‘mental arithmetic’, and ‘secure software’.

While the first two approaches yield a provably high strength against Trojan horse attacks they are expensive, difficult to use, and not flexible in regard to the data that they are able to sign. The latter ‘secure software’ approach usually disregards Trojan horse attacks in current implementations. Nearly all surveyed products did not bother to include protective measures against malicious processes on the same computer. On top they are difficult to use and inflexible. All three existing approaches force the signatory to explicitly review the data that is going to be signed before it is processed. The user must do this even if she just finished working with it in her application software. This sharply reduces the ease-of-use.

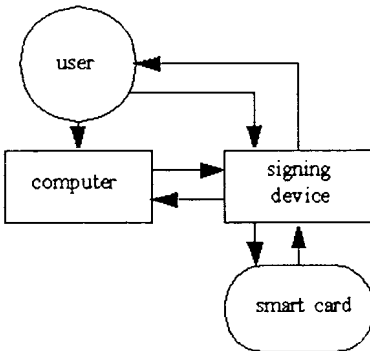
On the Cardis 2000 conference on smart card research and advanced applications Balacheff et al. proposed a technology to use smart cards in a partly secure environment. Their paper combined an ‘Intelligent Adjunct Model’ with a ‘Trusted Computing Platform’ approach pursued by some 150 hardware and software manufacturers. We show that a combination of the ‘secure software’ approach and Balacheff’s ‘Intelligent Adjuncts’ on a ‘Trusted Computing Platform’ can be used to protect the creation of digital signatures against Trojan horse programs on a computer.

2.1 Using secure hardware devices

This approach is favoured by most academic institutions and pursued by Cryptovision GmbH of Germany. They propose a device that comprises a liquid crystal display (LCD), a smart card reader, and a certified circuit board that contains the operating logic for the signing device; costs are estimated to be less than five thousand Euro each. The flow of information is shown in the sketch.

The computer is used as the provider of the information that shall be signed. Since the computer is assumed as completely insecure the signing device does not get the correct data if this has been manipulated by a Trojan horse on the computer. The signing device and the signature smart card are the only trustworthy components in the eye of the user. So the user has to review the data the signing device received to verify that it is the data she indeed wants to sign.

The data is displayed in a standardized way, eg, rich text format or common word processor file formats without advanced options. This implies that the presentation on the signing device does not always match the presentation on the signatory's computer. The receiver of the signed data may need the same signing device or a software viewer that displays the data like the signing device does. The signatory either confirms or rejects the presentation. After confirmation the data is sent to the signature smart card that actually computes the signature. The signature is then being transmitted through the signing device to the user's computer.



A Trojan horse that modifies the data before it reaches the signature smart card is detected by the user because she will note any modification of the data. Since the presentation and confirmation take place on a trustworthy device the signature is computed for exactly the data the signatory wants to get signed. The weakest component in this approach is a lazy user who does not review the data for reasons of convenience.

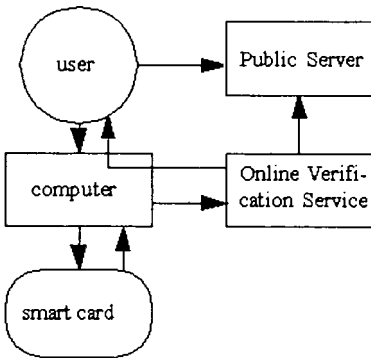
Drawbacks of the system are the small (and fixed) number of accepted file formats, the restriction to data that can be displayed on an LCD (eg, no audio data), the high costs that exceed the price for most personal computers, the high expenditure to roll out updates, and the reduced ease-of-use that diminishes the understanding and the support by the prospective users.

2.2 Neglecting arithmetically-challenged users

To avoid the high costs of additional hardware on the signatory's side T. Stabell-Kulø introduced an approach that forces the signatory to compute simple cryptographical operations by mental arithmetic. The user's computer is regarded as completely insecure and thus can not be trusted. A Trojan horse can alter every communication between the user and components connected to the computer. So Stabell-Kulø proposes to introduce an 'On-

line Verification Service' and a 'Public Server' to the PKI, and a One-Time-Pad and a substitution table for the user.

A signature computed by the smart card is sent to the (trusted) On-line Verification Service. This service verifies that the signature matches the data for which it was computed. It then encrypts the data by applying the substitution table and the One-Time-Pad and transmits this encrypted information through the insecure computer to the user; the signature is sent to the Public Server and marked as 'not yet released'. The signatory applies the One-Time-Pad and the substitution table to the encrypted information and retrieves the decrypted data. She compares if the decrypted data matches the data she wanted to sign. Decryption is done without the untrustworthy computer and takes approximately one minute for 15 characters if the user is a computer science student; otherwise the decryption rate averages 7.5 characters per minute. We did not find a signing method that has a lower ease-of-use. If the user agrees with the signed data she sends a release command to the Public Server that involves the use of a hash value.



While the advantage of obtaining a reliably computed signature in an insecure environment is not bad, the drawbacks are clearly disenchanting: the volume that can be signed is very low, the approach is not suitable for arithmetically-challenged people, you have to produce and securely distribute One-Time-Pads and substitution tables, and you have to introduce an On-line-Verification-Service and a Public Server to the Public Key Infrastructure.

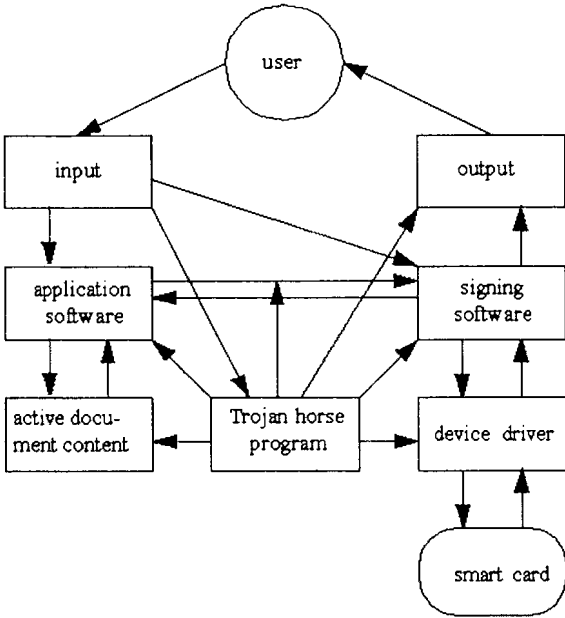
2.3 Implementing 'secure' software

This is the most common approach found in commercial products that are already being shipped to customers. We found that even leading companies disregard the threats posed by Trojan horses. One of the presumed market leaders was vulnerable to basic attacks. When asked why they did not use protection against Trojan horses they responded that it lies in the responsibility of the user to avoid the execution of untrustworthy processes on her computer. That is simply not suitable for a product used by inexperienced people.

The user works with an application software and at some point decides to sign the data. The data is then transmitted from the application software to the signing software, displayed for confirmation by the signatory, and finally sent to the signature smart card that computes the signature.

In most implementations a Trojan horse program has many interfaces it can attack: inside the application software as active document content, between the application software and the signing software, the signing software itself can be a target as well as the device driver between signing software and smart card.

The signing software displays the data that is going to be signed in a standardized way and prompts the user for confirmation. All but a few manufacturers assume that Trojan horses will not attack their software or explicitly put the responsibility for a Trojan horse-free environment in the signatory's domain.



In contrast to the first two approaches this is a low-cost approach without additional hardware devices or PKI components. However, it still has a low ease-of-use since the data has to be reviewed once more even

if the user has worked with it for a long time in the application software. And no company has made efforts to effectively block Trojan horse attacks on a conceptual basis.

2.4 Intelligent Adjunct model

In the Intelligent Adjunct model, the adjunct (ie, the smart card) is given control over off-card resources. The card can use all the resources, like peripheral components, the terminal it is connected to can use. Unlike former models that viewed the smart card as a passive component, in the Intelligent Adjunct approach a smart card can initiate transactions.

Connections to peripheral components are channelled through an Intelligent Adjunct service provider. This provider can be integrated in existing standards, eg, the PC/SC standard for accessing smart cards and terminals on the Microsoft Windows platform. Details of the implementation of the Intelligent Adjunct model are provided by Balacheff et al.

2.5 Trusted PC platform

The Trusted Computing Platform Alliance is an industry work group focused on enhancing trust and security on computer platforms. They have developed a specification for verifying the integrity of computer components. The goal is to build secure software applications on top of a verified computing base without the need of additional expensive hardware.

Starting from a hardware module that is resistant against software attacks, the components involved in the boot process of a computer are verified whether their integrity is untouched. The results of these measurements can be reliably retrieved by software that needs certain components to be trustworthy.

At the time this paper was written the Trusted Computing Platform Alliance had almost 150 member companies, including major industry players.

3. PROTECTING THE INTEGRITY OF THE SIGNING PROCESS

We combine the secure software approach with the Trusted PC platform. A negligence of the current secure software implementations is the lack of a check whether the peripheral components that are used can be trusted. We check the integrity of these components employing the integrity measures of the Trusted PC platform initiative. Thus we can rely on a safe input and output assuming that our PC's components do not include a Trojan horse on time of delivery by the clearly identified manufacturer who provides the integrity metrics.

We will elaborate on three scenarios regarding the storage of the signing software. The first scenario is the classic implementation found on computers today; the software is installed on the PC and stored on its local hard disk. The second scenario involves a Java virtual machine (JVM) that receives the code of the signing software from the smart card. This reduces the need of the smart card to verify the integrity of the installed signing software. It only needs to establish trust in the operating system which can include the JVM as a part. Our third scenario eliminates the need of a trusted JVM in the operating system. The signing software is executed on the smart card itself and communicates with the user by use of the intelligent adjunct model.

3.1 Verifying integrity of vital components

The Trusted Computing Platform Alliance (TCPA) proposes to introduce a so-called Trusted Platform Module (TPM) to a PC. It is a hardware component that can not be subverted by software attacks. It acts as the root of the integrity verification process.

When the computer is turned on the TPM verifies that the PC's BIOS conforms to the TCPA specification and thus can be trusted. The BIOS then verifies the operating system loader, the operating system loader verifies the operating system, the operating system verifies its JVM and so on. While the verification of the integrity of the operating system poses a challenge still not solved we are confident that this task will be addressed by the TCPA's member Microsoft.

Software running on a system with a TPM can request the results of integrity measurements to decide which components it will trust. As long as the path from the TPM to the software consists of TCPA-compliant components the software can rely on a trusted computing platform. This does not imply that the platform is free of Trojan horse programs. But it provides the means to identify the manufacturer of a Trojan horse since modifications of third parties can be detected.

3.2 Secure execution of the signing software

The signing software is usually installed on the signatory's computer once and onwards executed by loading it from the local hard disk. To avoid a Trojan horse attack on the signing software we have to check the integrity of the installed software before execution. We propose to store the software on the signature smart card itself. Since the functionality of the software can be kept limited we believe it would be feasible even with today's smart cards' capabilities. The software would be loaded to the operating system's JVM after its integrity has been verified by the card.

A third way would be to execute the signing software on the card itself. Suppose the signature smart card is a JavaCard we already have a trusted JVM available on the card. With the intelligent adjunct model the card gains access to the components needed to interact with the user. The card has previously established trust in the components by issuing a challenge to the TPM to check integrity of the components it is going to use.

Theoretically, a fourth way comes into mind. The signing software could be stored on the local hard disk and be executed in the signature smart card. We do not see advantages in this approach in contrast to storing the software on the card. If we have to load it to the card's JVM anyway, we prefer to store it on the card.

3.3 Trusted input

In the same way that one usually does not sign a blank piece of paper and let someone fill in the text at a later date, without having control, we do this on a computer system. The document that will be signed is being worked on with some application software, eg, a word processor. This application software may not be trustworthy and may represent a Trojan horse program. A big help for a Trojan horse is the labelling of the document as ‘to be signed’. Thus it is identified and becomes an interesting target for modification by an attacker.

We withhold this information from a potential Trojan horse by not allowing a direct connection of application and signing software. It may seem convenient to just click ‘sign’ in your application software, but it reveals the purpose of the document you work with. Instead, every application software has a standard ‘save work’ function that stores a document for further processing without determining the exact purpose. A document could be opened by the same user with the same application or a different one, by another user on another machine, it could be stored on backup media. In order to not risk being detected a Trojan horse program must abstain from altering a document that is just saved. Thereby we get an unmodified document that we now will transfer unmodified to the signing software.

This approach works with attackers who want to modify specific documents and benefit directly from the signatory signing these data. The assumption does not hold for attackers with the sole aim of vandalism. Here it would be useful to verify the file’s contents with a secure viewer, hardened against Trojan horse programs. We state below that the file is protected against modifications after saving and hence can not be altered by a Trojan horse program then.

Our method to ensure a reliable transfer to the signing software is not complicated. We use a WORM medium (Write Once Read Multiple). This medium does not allow modifications of a file after the file is closed. The file can only be read but not modified or deleted. A WORM can be implemented and details on the implementation are presented in an earlier paper (Cremers et. al. (2001)). For reasons of cost-efficiency we prefer a SWORM (=Software WORM), that does not require additional hardware. In the Trusted PC concept we hide the implementation and give the intelligent adjunct access to this secure storage.

Once we have stored the document to be signed in the SWORM as a trusted source we can use it to apply the signature to it. Some signature laws require that the user must be provided with another possibility to display the data the signature is applied to immediately before the signature creation.

This is perfectly possible with our solution. The presentation of the data can be done by opening the read-only file with the application software or by using a different software with the single purpose of displaying documents before signing.

The signing software opens the file stored on the SWORM and sends it to the signature smart card using a cryptographically-secured channel. The smart card is tamper-resistant and computes the signature for the data sent to it with the secret signing key of the signatory. To prove that the signatory is present the card usually requires a PIN input from the user. A secure input can be achieved by using a cheap smart card reader with an attached dedicated keyboard (eg, Cherry G81-8015).

3.4 Reliable presentation of signed data

We have stated that the data that is signed has no semantics in itself. A signed paper document can be interpreted by the parties involved and by a third party, eg, a court. You have a fixed presentation that is not altered by the way you look at it. In computer systems we have to use application software to give the data a meaning. The same binary data will be interpreted more or less different by different software products. While the same binary data has been signed by the smart card the interpretation of the signed document can be different at the site of the receiver without modifying the signed document itself.

A Trojan horse that is transferred with the signed document as active document content, eg, a macro, can alter the presentation on the receiver's computer. So the receiver may have an otherwise Trojan-free system but will nevertheless get a presentation not intended by the signatory. The simplest solution would be to disable active document contents at all. But this may reduce flexibility way too much.

There are two ways we propose to tackle this problem. The first way is a softer approach than just disabling active content. It should be possible to restrict the actions of active contents. So the receiver of a signed document should be able to determine which actions active content could perform on the document that would not alter the semantics. This requires cooperation of application software manufacturers regarding these options for their products. Today you can usually choose between allowing a macro to perform all actions or none.

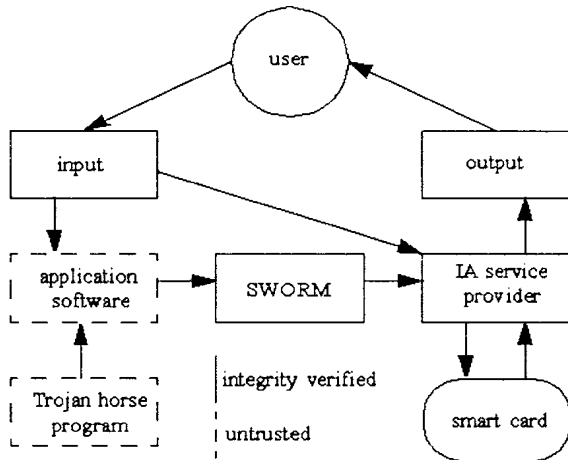
The other more promising method employs that a computer is a deterministic machine. So it is in principle capable of presenting the signed document in the same way as it has been on the signatory's computer. This can be done without cooperation of the application software manufacturers. The idea is to build a "sandbox" around the application that presents the

signed document. This sandbox sets all environment parameters that can be determined by active document content inside the application software and be used against a deterministic presentation. The environment parameters are collected at the signatory’s computer and include user name, computer name, network address, application software parameters etc. All parameters are included in an enhanced signature of the document so they can be evaluated by the sandbox on the receiver’s computer. The same parameters will lead to the same presentation. This sandbox approach is especially suitable for the verification step.

Sandbox software products already exist. Those tools usually focus on limiting the capabilities of potentially malicious programs by denying access to resources. They could be modified to simulate an environment similar to the one in which the signing took place.

4. IMPLEMENTATION OF A SECURE SIGNING METHOD IN SOFTWARE

Our method identifies components of the computer for which we have verified integrity and thus trust them to behave in the expected manner.



The document the user is going to sign is prepared in application software we do not trust. We transfer the document with a standard operation to a SWORM medium so a Trojan horse will not attempt to alter the data before it is signed. Transfer from the SWORM to the signature smart card to compute the signature can be achieved by one of three techniques depending on the place of storage and execution of the responsible signing software. Traditionally, software stored on a hard disk is used, but we encourage the

use of software stored as a Java application on the signature smart card. The software is then loaded to a verified JVM on the PC or executed in the card's JVM, communicating by Intelligent Adjunct technology with the outside world.

4.1 Trusted PC components

In the Trusted Computing Platform approach there is a path starting at the TPM along which the integrity of the components is verified. For our method this path must lead from the TPM to the operating system and must include input and output peripheral components, the SWORM medium, and the Intelligent Adjunct service provider.

The application software the user runs to create and modify her documents does not have to be TCPA-compliant. We avoid Trojan horse interference by reducing crucial information for an attacker and securing an early input as described in sections 4.4 and 4.6. Vandalism is still possible to a certain extent.

4.2 Intelligent Adjunct functions

The Intelligent Adjunct service provider (IASP) coordinates communication between the signature smart card and off-card resources. An IASP will usually be part of the operating system and thus its integrity will have been verified. If it is not part of the operating system, its manufacturer has to provide TCPA-compliant integrity information.

Components that are addressed by way of the IASP are the Trusted Platform Module, input and output devices, the SWORM medium, and the JVM of the operating system.

4.3 Signing software stored on signature JavaCard

Obviating the need of secure storage of signature software on a signatory's computer is clearly an advantage. Even with TCPA technology the software manufacturer has to provide TCPA-compliant integrity information with each software update. By use of a JavaCard as secure software storage we gain higher security and platform-independence. Execution of the software occurs on a verified JVM and the application uses only components of which the integrity has been verified beforehand.

The signature smart card manufacturer can provide the signatory with a platform-independent solution for signing digital documents. The application is integrated on a single smart card and can be used with every terminal that

implements the proposed methods, regardless whether it is the user's own computer or a public signing terminal.

As long as current signature smart cards are not capable of executing the signature smart card in their own on-card JVM, the software has to be transferred to the (verifiably trustworthy) JVM on the terminal's side.

4.4 Early unchangeable input for signature creation

The proposed use of a SWORM is central to our concept since it is now possible and feasible to keep the input for the signature fixed at a very early stage. In contrast to other approaches we get the input at the earliest time possible before a Trojan horse even knows that the document will be signed and thus become interesting.

To achieve a high level of security, the implementation of the SWORM should look like a non-SWORM medium. This prevents active document content from testing if the target of a standard 'save work' command is a SWORM medium

Details on how we implement the SWORM medium can be found in an earlier paper (Cremers et. al. (2001)).

4.5 Exact data labelling for deterministic presentation

The proposed sandbox method requires that we gather as much input parameters for a deterministic presentation as possible. Since we do not rely on cooperation with the application software manufacturers we have to pick up the parameters at various places.

We think that the following parameters must be included with the signature to make a possible Trojan horse on the receiver's side believe it is executed on the signatory's machine. These parameters are the document format, the application used for working with the document, the version of the application, the parameters for the application (stored in the Windows system registry or in a configuration file), the size and colour depth of the Windows desktop, available fonts on the system, information about the origin and integrity of the fonts, the user name, the machine name, the network address, number and labels of storage media, serial numbers of the machine and application.

The sandbox is built around the application used to present the signed document. Since not every program is capable of being run reliably in a sandbox it may prove necessary to disable active document content entirely on the system in those situations.

Current sandbox software products operate on a simple allow/deny-basis for access to information and computing resources. They have to be

enhanced to provide applications with (misleading) information about the environment in which applications are executed.

4.6 Reducing valuable information for corrupt participants

We rise the risk of detection for a Trojan horse by withholding information it needs for a successful attack. The document is transferred to the SWORM medium without determining the purpose of the transfer. A Trojan horse that modifies the document anyway when it is saved takes a high risk in being detected. By definition, a Trojan horse program has to keep its existence a secret, so it will not risk being detected. Otherwise the origin of it could be traced and the attacker be held liable in a court. So even if a Trojan horse resides on the signatory's system it is not able to catch the right time to interfere with the signing process.

The reduction in information for an attacker is enforced by a user policy. A signatory is advised not to use seamlessly-integrated plug-ins in her application to trigger a signing process. Instead it is necessary to direct the output of the application to the SWORM and making it look like a standard and not suspicious action.

5. CONCLUSION

Trojan horse programs (THP), ie, programs with additional hidden, often malicious, functions, are more and more popular forms of attack. On the one hand, high-level macro programming languages in many office applications make it easy even for inexperienced attackers to write, hide and distribute a THP. On the other, the emerging digital signatures are likely to become a favourite target of attacks. The owner of a digital signature can face considerable damage if a THP is able to sign a document the user did not intend to.

By identifying components of the signatory's computer of which we can check integrity we assemble a trusted computing base. We use this base to execute the signing software in a safe environment.

First of all the signatory determines the data that she wants to sign in the application software way ahead of the actual computation of the signature. It is not necessary to perform an additional presentation of the data in the signing software. Second the transfer of the data through the signing software to the signature smart card is secured by the Trusted PC platform.

Third the solution can be achieved without additional expensive hardware, thus lowering the financial burden of security.

Active document contents still pose a special problem if a user is not willing or able to disable their execution. To ensure that a document is displayed identically on both the sender's and receiver's display, both parties must run the same document processing program with the same profile, ie, the same command-line parameters, options etc. Thus, in addition to a document and its signature, the sender must include the name, version and used profile of her document processing program, which the receiver must use to view the document. This is supported by using a sandbox approach in the signature verification step.

In conclusion, we have shown that the threat of Trojan horse programs attacking a document's integrity can be averted with only a few measures, which – compared with previous approaches – retain a system's flexibility and incur only minor inconveniences on its usability.

References

- Balacheff, B., D. Chan, L. Chen, S. Pearson and G. Proudler (2000). 'Securing Intelligent Adjuncts Using Trusted Computing Platform Technology'. *IFIP TC8/WG 8.8 4th Working Conference on Smart Card Research and Advanced Applications*. pp: 177-195.
- Bontchev, V. (1996). 'Possible macro virus attacks and how to prevent them'. *Computers & Security* 15(1996):595-626.
- CERT Coordination Center (1999). *CERT Advisory CA-99-02-Trojan-Horses*.
<http://www.cert.org/advisories/CA-99-02-Trojan-Horses.html>
- Cremers, A. B., A. Spalka and H. Langweg (2001). 'Vermeidung und Abwehr von Angriffen Trojanischer Pferde auf Digitale Signaturen'. 7. *Deutscher IT-Sicherheitskongress*. Bonn, May 2001 [German]
- Docherty, P., and P. Simpson (1999). 'Macro Attacks: What Next After Melissa?'. *Computers & Security* 18(1999):391-395.
- European Parliament and European Council (1998). 'Directive on a Community framework for electronic signatures'. C5-0026/99 - 1998/0191 - (COD).
- Ford, R. (1999). 'Malware: Troy Revisited'. *Computers & Security* 18(1999):105-108.
- Hoffmeister, A., Cryptovision GmbH (2000). Personal communication.
- Lacoste, G. , B. Pfitzmann, M. Steiner and M. Waidner, ed. (2000) *SEMPER – Secure Electronic Marketplace for Europe*. Berlin et al: Springer-Verlag.
- Lapid, Y., N. Ahituv and S. Neumann (1986). 'Approaches to Handling "Trojan Horse" Threats'. *Computers & Security* 5(1986):251-256.
- Popek, G.J., and C.S. Kline (1977). 'Encryption Protocols, Public Key Algorithms and Digital Signatures in Computer Networks'. R. A. DeMillo (1978). *Foundations of Secure Computation*. pp:133-153.
- Pordesch, U. (2000). 'Der fehlende Nachweis der Präsentation signierter Daten'. *DuD Datenschutz und Datensicherheit* 24.2(2000):89-95. [German]
- Schmidt, A. U. (2000). 'Signiertes XML und das Präsentationsproblem'. *DuD Datenschutz und Datensicherheit* 24.3(2000):153-158. [German]

- Spalko, A., A. B. Cremers and H. Langweg (2001). 'The Fairy Tale of "What You See Is What You Sign"'. Trojan Horse Attacks on Software for Digital Signatures'. *IFIP Working Conference on Security and Control of IT in Society-II*. Bratislava, June 2001.
- Stabell-Kulø, T. (2000). 'Smartcards: how to put them to use in a user-centric system'. *HUC2K The Second International Symposium on Handheld and Ubiquitous Computing*. <http://www.pasta.cs.uit.no/publications/HUC2K.html>.
- Trusted Computing Platform Alliance (2000). *TCPA Trusted Subsystem Specification Version 0.9*. <http://www.trustedpc.org>.

This Page Intentionally Left Blank

Security Concerns for Contemporary Development Practices

*A Case Study**

T. Tryfonas and E. Kiountouzis

Dept. of Informatics, Athens University of Economics and Business

76 Patission St., GR-10434, Athens, HELLAS

{tryfonas, eak}@aueb.gr

Key words: Action Research, Contemporary IS Development Approaches, Information Systems Security Design, Information Systems Security Practices

Abstract: This paper presents a case of application of an interpretive framework, which intends to formally integrate information systems security concerns within the information system's lifecycle. Aspects that are not normally taken under consideration, such as the involved stakeholders, the development approach and their implication to security issues, are introduced in such a way to benefit and empower the IS security design process. In the case presented here, the framework is used to extract a powerful process model description focusing on security concerns, so as to enlighten the work of the security designer significantly earlier before the use of risk analysis and the construct of a security plan or policy.

1. INTRODUCTION

New forms of communicating and trading require a robust and secure technical infrastructure in order to be performed and be fully exploited. In modern organisations assets can often be found in the form of data stored, processed and transmitted by information technology (IT) facilities, in the form of products, systems or applications; such data is a critical resource that

* Supported in part through YPER97 programme by the Hellenic Ministry of Development.

enables organisations to succeed in their mission. Stakeholders of those assets often require that dissemination and modification of any such information representations are properly controlled, as organisations or individuals have a reasonable expectation that their data remain private, be available to them as needed, and not be subject to unauthorized modification; this might stem from organisational requirements (company's regulations, organizational policies) or environmental necessity (market trends, data protection acts). Thus, there is an expectation from IT management to facilitate the identification and implementation of security controls to ensure that data are protected against potential threats. Traditionally, such security controls come up as a risk analysis (RA) result. This is a particularly effort-consuming process and it is usually performed after the information system's (IS) development. When applying such a practice, an "instance" (model) of the existing IS needs to be extracted, a fact that requires study of the system's structure and processes and constant contact with experienced users and designers. From this picture security specialists shall try to evaluate the organizational information assets, whilst at the same time they must validate each assessment with the system's stakeholders.

There are many RA approaches for system's security, most of which based upon the scientific paradigm, and are applicable along with similar development approaches. For example Downs, Clare and Coe (1992) study the relation between the CCTA Risk Analysis and Management Method (CRAMM) and the SSADM development methodology. They say that the former has been designed in a way that could be exploited in every development project that uses SSADM-like rationale. The technique that they imply is quite simple and has to do with the enrichment of the system's requirements ("Requirements Catalogue"), with the security requirements regarding the organizational assets, as they emerged through CRAMM's Stage-1. Further studies address this topic and present in detail the SSADM/CRAMM interface (Baskerville, 1993).

On the other hand, modern technology solutions are procured and developed in ways that they make extensive use of existing commodity IT products (i.e. hardware, operating systems, middleware, general-purpose applications, ERP systems, communication services etc.). Several developers

and users of IT solutions lack the knowledge, expertise, or resources which are necessary in order to judge whether they may trust the security level of their IT components. In addition, traditional RA techniques are not particularly successful when facing such requirements that stem from the way contemporary systems are developed.

In this context, modern information systems should perform their functions whilst ensuring information protection against hazards such as unwanted or unwarranted dissemination, alteration, or loss. The existing definitions for information security as the protection of confidentiality-integrity-availability, or IT security as the protection of the computational infrastructure, do not take under consideration “soft” factors within an IS, such as the human account, legislation, market requirements etc. There is a need for a comprehensive, systemic approach capable to resolve IS security issues, taking into account such important factors that cannot be traced through traditional practices. In the light of that assumption, the integration of systems security with modern IS development practices could be used to prevent and mitigate the previous or similar hazards. In the complex context of an IS, combining people, information, software, hardware and procedures, information technology security cannot ensure by itself the security of the entire system. IS security is indeed a broader term, containing all principles, regulations, methodologies, techniques and tools we establish and use to protect an IS, or any of its parts, from potential threats.

This paper validates a conceptual framework capable of resolving security problems within systems development; we briefly summarize the systemic approaches for IS security (section 2), we present an interpretive framework constructed to exploit such approaches and empower the IS security design (section 3), we introduce our research method (section 4) and eventually the case under study (section 5).

2. A BRIEF SUMMARY OF PREVIOUS RESEARCH

As an IS introduces managerial problems to the lifecycle of an organization, IS security problems should therefore be considered to be managerial and therefore be approached through problem solving

techniques. Successful resolution is the procedure where a carefully designed and planned change takes place within an organizational context and becomes the body of relation between the present reality (problematic situation) with the desired ending (designed situation). Mumford (1998) sustains that action research is the most suitable way of resolving managerial problems, as the production of good theory cannot be done in isolation and without the involvement of the researcher to the organizational problems. Kiountouzis and Kokolakis (1996) argue that regarding information systems development there is a constant transformation of the analyst's way of thinking from the systemic to the systematic paradigm and vice versa, as like walking onto a Moebius band. The understanding and analysis of the problems is achieved by a systemic way, whilst the design and the implementation of the solutions through a well defined systematic way; thereafter the evaluation and assessment of the solution that shall lead to a possible success, failure or an IS redesign is achieved through systemic procedures, so as to document new requirements and start this process all over again. Similarly, IS security problems can be resolved only if the practitioners could continuously change their way of thinking from systematic to systemic and vice versa.

Existent methodologies for IS development do not meet the needs for resolving the security-related IS problems as most of them neither do include specialized handling of the security requirements nor can create a design for security controls early in the development process. In addition there are not many adequate studies for the exploitation of existing techniques and tools that could contribute to the formal and convenient integration of the security requirements within the IS development requirements (Hitchings, 1995a). Furthermore, the existing formalisation attempts (models, mathematical foundation) are limited in scope and cannot capture either in detail or in a comprehensive way the dynamics of the security concerns of contemporary information systems within the context of modern organisations and the technological progress (Kokolakis, 1996).

To cope with this Hitchings proposed a systemic theory for IS security design (Hitchings, 1995b); in particular her work, the *virtual methodology* (VM), along with the risk analysis method *Security By Analysis* (SBA) are two of the few systemic approaches for systems security. Such approaches

seem to be the appropriate way of resolving problems, as within information systems “soft” factors (ethics, legislation, training, familiarization, user satisfaction etc.) are of major importance for its security and protection.

The VM takes under consideration the dynamic nature of the IS and the variations and uncertainty that the human factor introduces to it. Using the technique of consensus as used in the Soft Systems Methodology (SSM) (Checkland, 1981), where the establishment of designs is achieved upon agreement of the involved stakeholders, an organisation model is firstly introduced, then the IS model that corresponds to it and finally risks against them are defined. From that point and thereafter one can build up a control environment tailored to the particular organisational and informational needs.

A clearly systemic approach as well is the SBA risk analysis method. It was developed in Sweden and became an RA standard; it reflects the Scandinavian culture and attitude towards security aspects, as it is a product of an environment where the socio-technical design of systems flourishes. It is based on a thorough review of security breach scenarios, as they are identified through meetings amongst the various stakeholders. The control environment set-up comes through mutual agreements and consensus about what is important for the system (asset valuation) and what risks are there for those assets (risk assessment).

This approach tries to encapsulate all factors that have a potential security impact, as the meetings include every stakeholder (analysts, end-users, management etc.) and therefore each single perspective of the CATWOE analysis is included (Checkland, 1981). The previous term is an acronym referring to: the Customers of the system, the Actors involved, the Transformation of the input to the desired output, the Worldview under which the transformation has a meaning, the Owners of the system and the Environment. Ideally through that practice the stakeholders shall pinpoint the assets and the risks against them so as to design security countermeasures. The problem of such participatory approaches is that they need to be guided by extremely skilled analysts that shall act as facilitators of the entire process.

Information security problems in contemporary product/component oriented development practices could be resolved in the broader context of

QA assurance, since each single product could be validated and assured properly. Eloff and Von Solms propose a holistic approach based on the validation of products along with process assurance, combining both system components and system processes (Eloff and Von Solms, 2000). They argue that lack of quality assurance in IT production is the heart of the problem. However, assurance of high-quality development cannot by itself ensure security, as even the perfect product could be a subject to misuse.

Finally, Ynström (1999) proposes a systemic-holistic framework to approach IT security exploiting systems principles, control theory, SSM and cybernetics. Such theoretical frameworks are largely suitable for the resolution of security problems that are not strictly technical, but also other systemic (“soft”), factors could have an important role in a problematic situation.

3. THE NEED FOR A NEW FRAMEWORK

In the light of the previous systemic approaches information systems security design is viewed as the process that includes all tasks that aim to establish a mature level of security and protection for the information system as a dynamic comprehensive organisational subsystem. When designing a security plan for an organisation it is very important to rely upon a “rich picture” of the problem, i.e. an appropriate organisational model that shall help analysts to understand processes and focus on potentially problematic areas in designing safeguards. This model’s operation is twofold as:

- On the one hand it is a means for validating common perceptions amongst the various stakeholders, and as a result the analysts can have a complete description of the organization.
- On the other hand it is the basis for the safeguards design, as it helps in tracing all deficiencies, potential risks etc.

Security design should be accomplished through disciplined ways, as this is a practice to ensure reusability of the results and of the gained experience and knowledge and have guidelines of how to deploy them in the future. The use of methodologies for system development and related issues (and therefore security) is necessary indeed because of the

complexity and volume of such projects. There are strong arguments for choosing to do so, such as (Fitzgerald, 1998):

- The analysis of complex processes to easier to-be-handled sub-areas.
- The facilitation of project management.
- The reduction of the total uncertainty and of potential risks against the project.
- The facilitation of standardisation processes and repeatability of the method.

But many projects are implemented virtually without any methodological support due to limitations, such as:

- The great number of methodologies to choose from and an important lack of standardization of tools, deliverables and products.
- The fact that many are generalizations of theoretical or empirical research work and have not been properly validated with regard to whether they can be efficient and effective to other projects as well.

The information systems literature, in which the methodologies movement flourished in the eighties and early nineties, has not addressed sufficiently the new norms of practice and there should be introduced a classification of contemporary systems development practices along the well-known ‘make or buy’ divide (Tryfonas et al, 2000). Most systems projects are now anchored on the ‘buy’ maxim; on that, two development approaches are introduced namely the *single-product based* and *component-based* development. On the ‘make’ side we have proprietary development. Each of these three approaches introduces different challenges to developers, consultants and users (considered to be the high-level involved stakeholders in the framework, introduced in Table 1) regarding security concerns. A stakeholder is anyone involved in the situation that could gain benefits from it (Pouloudi, 1999).

Table 1: Integration of IS development with security issues (Tryfonas et al, 2000).

		IS SECURITY					
		ABSTRACTION LEVEL			EXPRESSION MODE		
		<i>Strategy</i> (Appear in corporate strategic plan)	<i>Design</i> (Embedded in development practices/methodologies adopted)	<i>Implementation</i> (Embedded in acquired technology and products)	<i>Explicitly declared security concerns</i>	<i>Implicit security understanding («metaphor»)</i> – Security is ...	
IS DEVELOPMENT	APPROACH	Product-based	All security concerns that appear and influence an organisation's planning; policies etc.	Securing techniques and methods that accompany systems development practices	Security tools and specific technologies, which are appropriate to be used when a particular method is selected.	All issues declared to be concerned with systems security within development approaches.	Security issues implied by the epistemology and the discipline followed (e.g. cultural concerns)
		Component-based					
		Custom/Proprietary					
	STAKEHOLDERS	User organization	Perception of the stakeholder for IS security in strategic level	Perception of the stakeholder for IS security in design level.	Perception of the stakeholder for IS security in implementation level	All declared statements of referenced organisations about security: describe their understanding of the subject.	All implied concepts concerning security within the referenced organisations.
		Developer organization					
		Consultant organization					
	Environment	Environmental influence on IS Security strategies.	Environmental influence on IS Security Design	Environmental influence on countermeasures implementation and use.	Explicitly recognized environmental influences (e.g. data protection acts).	Implicit influential parameters (market trends, user satisfaction etc.).	

This interpretive framework is to be applied in development/research projects. Such kind of verification and redesign is a proper way of approaching organisational research problems related to information systems because it can collaborate the theoretical solution design phase with its practical application and evaluation through practice (Checkland, 1999). Systems theory and organisation theory for the inspection of the relation between the organisation, its IS and the IS Security, are very suitable

approaches to this action research paradigm. These ideas along with SSM constitute the foundations of this theoretical framework. Those tools do not only resolve the problem of what technology to use and how to use it, but also address who require the solution, which get benefited from it and what could be the social, legal and political impacts of a design.

4. RESEARCH APPROACH AND METHOD OF INTERVENTION

An appropriate way of validating an interpretive theoretical design is by applying it to real-world cases and eventually refining it based on findings and gained experiences (Walsham, 1995). In general that kind of research (action research) includes active application of solutions/proposals and is a process that leads from practice to corresponding theory and vice versa (Eden and Huxham, 1996). This circular process begins with the informal understanding of the problem that leads to formal documentation of the theory that leads in consequence to resolution actions that can be verified across other similar cases and eventually introduce a robust theoretical framework for facing the problem.

Basic characteristics of action research are (Klein and Myers, 1999):

- It is an iterative and incremental approach.
- Intervention of the researcher to the cases under study is necessary.
- It is always context dependent.
- It produces customer-centred solutions.
- It is a process where generalization without thorough repeatability evidence is a rough to resolve issue and the validation of results is achieved through successful application to similar cases.
- When successful it reflects theory to a robust formal system.
- The presentation of results and findings is rather loose and quite radical.

Those principles lead to research conduct that deals with the heart of problematic situations and when they are successful they lead to production of real and viable theory construction.

5. THE CASE UNDER STUDY

The goal of the case under study was the implementation of a comprehensive security plan for the IS of a non-governmental non-profit organisation, offering treatment programmes for addictive individuals. In detail this project aimed at:

1. Modelling and documenting all information systems supported processes, especially those related to sensitive data.
2. Conduct of risk analysis for the IS.
3. Development of a security policy for the organisation.
4. Documentation of all security countermeasures.

Objectives (3) and (4) comprise the Security Plan and were the project's final deliverable.

5.1 Organization profile

The organisation under study is responsible of running eight different programmes to support addictive individuals (mainly drug-addicts and partially alcoholics). Its programmes are distributed all over the country, based on major cities. Stakeholders of the organisation are:

- The management.
- The staff and the major end-users of the IS.
- Program members and their families.
- The Ministry of Health and Social Security Affairs.
- The Data Protection Authority.

The model we shall construct shall introduce and present in detail processes within the organization that utilize information and could have potentially implications by a security incident, as well as all services to end-users, of any kind (drug additives, researchers, therapists etc.). For the identification of security concerns depending on the type of the system (from the way it was developed point-of-view) and the involved stakeholders we shall combine the framework presented in Table 1 with a traditional risk analysis rationale (asset identification and valuation and potential risks against them) so as to facilitate the dissertation of the security plan.

The following sectors constitute concrete organization functions:

- Research sector,
- Treatment design and delivery sector,
- Administrative and financial services,
- Public relationships sector,
- Public awareness and forestalling unit,
- IS development and support sector, and
- Training sector.

5.2 Information system characteristics

The information system of the organization serves two major goals:

1. Financial and administrative support.
2. Support of research and of design of therapy, evaluation programmes and training.

To meet the second goal, processing of data considered to be sensitive (medical records, contact info of addictive individuals etc.) is required. Ensuring security of the sensitive data that the organization utilizes is a twofold necessity; on the one hand, legislation and the data protection act in Greece explicitly state that these data should be properly secured and on the other hand, trust is a major quality of such a kind of organization and therefore should be safeguarded at any cost.

The corresponding informational infrastructure for the organisational sectors and their processes shall be briefly introduced here; all workstations are interconnected PCs through two local area networks. One is for use by the financial administration services and one for use by the research sector. The information system consists of various independent applications and platforms that include an accounting package, office automation software, statistical analysis tools and custom applications. Those applications are hosted in a Windows NT/98/95 network (1 server, 22 w/s in the central building), 1 Novell network (accounting, 6 w/s), 1 independent Macintosh and four laptop computers.

Communication with third parties takes place through phone, fax and e-mail, the latter being provided by an Internet service provider through a dial-up connection. Informational support per process can be seen in detail in

Table 2, where we construct a security requirements-driven organisational process model.

5.3 Experiences and lessons learnt

Practitioners that try to resolve security problems of information systems should work their way towards it by developing common understanding between stakeholders by the use of and compliance with standards or other “disciplined” approaches (like risk analysis), a key role in which have:

- (a) *the efficient modelling of the organizational environment and*
- (b) *the proper exploitation of those models extracted.*

In the light of the previous argument, we could say that with regard to the case under study the proposed technique was rather suitable. The sensitive nature of the processed information and the organizational requirements were such that success could be achieved only through a systemic, disciplined way. There was an explicit statement of the need for a methodological support of the IS security design.

Modelling inscriptions of the processes and their association with security issues per development approach is a powerful tool in the process of security design because Representing the organizational activities, including models for an efficient description of involved roles and their corresponding perceptions and responsibilities and associating them with security issues, enlightens the security design at any time and especially facilitates the early integration of security concerns within the IS requirements/analysis and design phases.

Table 2: IS security requirements driven organisational process model.

Administrative and financial services	Windows and Novell w/s with <ul style="list-style-type: none"> • accounting (package), • payroll system (package), • balance-sheets administration (custom-made) 	management, staff	Financial data	<ul style="list-style-type: none"> • Corporate strategy compliance • Assurance of contracts • Control & audit • Security standards certification/compliance • Technology transfer (in-house) • Copyright protection (outsourced) • Concern for scientifically sound approach • Market acceptability
Training	Windows w/s with: <ul style="list-style-type: none"> • database applications (custom-made) 	management, staff	Employee personal data, employee evaluations	
Public relationships	Windows w/s for elementary word processing	management, staff	Press releases, journals, articles, announcements, newspapers etc.	
Research	Windows w/s with: <ul style="list-style-type: none"> • statistical analysis (package) 	management, staff ministry of health, data protection authority	“Anonymised” sensitive data of subjects partaking in drug-addiction programmes	
Treatment design and delivery	Windows w/s with: <ul style="list-style-type: none"> • full office automation applications (packaged solution) 	staff, programmes members, data protection authority	Sensitive information (personalized contact info, medical records, other information)	
Forestalling	Windows w/s for elementary word processing	staff, potential programmes members	Press releases, journals, published reports, articles, announcements etc	
				<ul style="list-style-type: none"> • Integration to system specifications & requirements • Non-functional requirements • Risk prioritisation • System modelling plus risk analysis
				<ul style="list-style-type: none"> • Product’s features configuration • System configuration features • Authenticity • Non-repudiation • Continuity of service • Access control • Authorizations

6. GENERALIZATION OF CASE EXPERIENCES

Development practices and scenarios vary, so do practices of IS security. In general we can identify the following high level counter-practices:

- analysis of the system under study and risk assessment,
- set-up of a security policy,
- assurance that it complies with industry standards/laws,

- verification of the policy's competence and the system's security level by repeatedly analysing the risks against it.

We argue that even before the security designers can benefit from risk analysis and the establishment and use of security policies, they could benefited very early in the IS development and security design process from particular domain analysis and modelling inscriptions (e.g. Table 2) so as to identify early the major security concerns.

Modern information systems in their contemporary organizational context (rapid technological progress, changing forms of applying business, changing norms of communications, informational added value to organizational processes) need now, more than ever, approaches to assure their security in a convenient, dynamic and effective way; the luxury of "enough time" to study the system and conduct risk analysis in the traditional way, over a static image of it, is not applicable any more and security design needs to centre around approaches that ensure early tracking of potentially problematic areas and their effective counteraction.

Approaches like the interpretive framework we presented in action through this essay, we believe that shall contribute to the empowerment of the process of information systems security design and that it shall make it possible to integrate security to the information system's development processes.

7. REFERENCES

- Baskerville, R. (1993), "Information Systems Security Design Methods: Implications for Information Systems Development", *ACM Computing Surveys*, Vol. 25 No. 4.
- Checkland, P. (1981), *Systems thinking, systems practice*, Wiley.
- Checkland, P. (1999), *SSM: a 30-year retrospective*, Wiley.
- Downs, E., Clare, P. and Coe, I. (1992), *SSADM: Application and Context*, Prentice Hall.
- Eden, C. and Huxham, C. (1996), "Action Research for the Study of Organizations", in *Handbook of Organization Studies*, S.R. Clegg, C. Hardu, W.R. Nord (eds), Sage.
- Eloff, M., and Von Solms, B. (2000), "Information Security: Process Evaluation and Product Evaluation", in *Information Security for Global Information*

- Infrastructures*, S. Qing and J. Eloff (Eds.), Kluwer Academic Publishers, pp. 11-19.
- Fitzgerald, B. (1998), "An empirical investigation into the adoption of system development methodologies", *Information & Management*, 34, pp. 317-328.
- Hitchings, J. (1995a), "Deficiencies of the Traditional Approach to Information Security and the Requirements for a New Methodology", *Computers & Security*, 14, pp. 377-383.
- Hitchings J. (1995b), "Achieving an Integrated Design: The Way Forward for Information Security", in *Information Security – the next decade*, J. Eloff and S. von Solms (Eds.), Chapman & Hall.
- Kiountouzis E.A. and Kokolakis S.A. (1996), "An analyst's view of IS Security", in *Information System Security facing the information society*, S. Katsikas and D. Gritzalis (Eds.), Chapman & Hall, pp. 23-33.
- Klein, H. and Myers, M. (1999), "A set of principles for conducting and evaluating interpretive field studies in information systems", *MIS Quarterly*, Vol. 23 No. 1, pp. 67-94.
- Kokolakis, S. (1996), "Is there a need for new information security models?", in *Communications and Multimedia Security II*, P. Horster (Ed.), Chapman & Hall.
- Mumford, E. (1998), "Problems, knowledge, solutions: solving complex problems", *Journal of Strategic Information Systems*, 7, pp. 255-269.
- Pouloudi, A. (1999), "Aspects of the Stakeholder Concept and their Implications for Information Systems Development", *Proceedings of the 32nd IEEE International Conference on System Sciences*.
- Tryfonas, T., Kiountouzis, E. and Poylimenakoy, A. (2000), "Embedding security practices in contemporary information systems development approaches", submitted to *Information Management and Computers security*.
- Walsham, G. (1995), "Interpretive case studies in IS research: nature and method", *European Journal of Information Systems*, 4, pp. 74-81.
- Ynström, L. (1999), "Systemic-Holistic Approach to IT Security", in *IPICS 99 lecture notes volume*, University of the Aegean.

This Page Intentionally Left Blank

A Paradigmatic Analysis of Conventional Approaches for Developing and Managing Secure IS

Implications for research and practice

MIKKO T. SIPONEN

University of Oulu, Department of Information Processing Science, Linnanmaa, P.O.BOX 3000, Oulu, FINLAND. E-mail: Mikko.T.Siponen@oulu.fi

Key words: Information Security Management

Abstract: Because the methods of development for Information Systems (IS) do not pay attention to security aspects, several information systems (ISS) security methods have been presented. This paper will analyze traditional/conventional approaches, namely normative standards (e.g. checklists, management and evaluation standards), formal methods, common sense principles and risk management. These approaches will be analyzed in the light of I) the research objectives; II) the organizational role of IS security; III) research approaches used; IV) applicability; and V) a conceptual meta-model for IS. The contribution of the paper is twofold. First the analysis sheds new light on the underlying foundations of the conventional approaches. Second, the analysis suggests several implications for researchers and practitioners.

1. INTRODUCTION

Despite the recognized relevance of IS security (e.g. Baskerville, 1992; Straub & Welke, 1998), IS security design aspects are neglected in IS development methods (Baskerville, 1992; Dhillon & Backhouse, 2001). The information security community at large has gotten stuck in technical small-scale questions (e.g. Dhillon & Backhouse, 2001; Thomas & Sandhu, 1994) and the thesis propounding that the key issue in development is “formalization” (e.g. Anderson, 1993; Barnes, 1998). To overcome this weakness, several methods for the development of secure ISs, from checklists to different approaches based on IS or software development

methods, are proposed (Baskerville, 1993; Dhillon & Backhouse, 2001; Siponen, 2001a). It is interesting to note that the naturalistic approaches, such as normative standards, risk management and formal methods, have survived well. Checklists (herein classified as normative standards) and risk management approaches have been widely used (Baskerville, 1992; Fitzgerald, 1993) and different normative standards such as Generally Accepted System Security Principles (GASP, 1999), SSE-CMM (1999a; 1999b) and BS 7799 (1993) have been recently announced. Furthermore, different normative standards are highly rated by many security experts (e.g. Fitzgerald, 1995; von Solms, 1997; 1998; 1999). Additionally, the use of formal methods have been advocated by the Computer Security community (e.g. Anderson, 1993; Barnes, 1998). Recently, an interest to scrutinize the theoretical foundations of the alternative approaches for designing and managing IS security have been increased. Consequently, Baskerville (1993), Parker (1998) and Siponen (2001b) have taken a critical look at checklists and security management standards. Dhillon & Backhouse (2001) and Dhillon (1997) have analyzed methods for developing secure IS's in the light of Burrell & Morgan. Siponen (2001a) have analyzed the recent (non-conventional) approaches. This paper continues these research efforts. The conventional approaches will be analyzed from the viewpoints of I) the research objectives; II) the organizational role of IS security; III) research approaches used; IV) applicability; and V) a conceptual meta-model for IS.

Conceptual analysis in terms of Järvinen (1997) is used as the research approach of this paper,

The rest of this paper is organized as follows. In the second section, the framework for analysis is presented. In the third section normative standards are analyzed. In the fourth section, the risk management approach is considered. Fifth section analyzes formal methods. The sixth section discusses the implications of this study. In the seventh section, the key issues are summarized.

2. THE FRAMEWORK FOR ANALYSIS

The following framework will be used to carry out the analysis.

Table 1. The viewpoints/tools of the analysis

Viewpoints	Reasons
1) The research objectives	1) To perceive what is the goal of research
2) Organizational role of Information Systems Security	2) To see what is the organizational role of IS security development

Viewpoints	Reasons
3) Research approaches used	3) To see what research approaches are used and preferred to a) develop IS security methods; b) validate the solutions
4) Applicability into IS and software development	4) To see whether the security methods can be integrated to IS or software development
5) A meta-model for IS	5) What aspects of IS do the contributions cover?

These viewpoints are discussed next.

The research objectives

Based on the classification by Chua (1986), the objective of scientists can be divided into 1) means-end oriented; 2) interpretive; 3) critical - though we shall herein simplify these concepts. A means-end oriented view holds that the aim of research is to produce knowledge for achieving certain concrete goals or ends. For example, development of a new algorithm is an example of means-oriented research. For Chua, the means "to enrich people's understanding of the meaning of their actions" (Chua, 1986 p. 615). The importance of interpretive research is widely accepted in social sciences – and IS science have close connections to social sciences since information systems are social systems (Hirschheim, 1985). Hence, interpretive research seems to be relevant for IS (e.g. Hirschheim, 1985; Walsham, 1996; Galliers & Swan, 1997; Klein & Myers, 1999). The goal of critical research is to point out the weaknesses of the existing theories/practices.

Organizational roles of Information Systems Security

Three organizational roles of Information Systems (Security) can be categorized into technical, socio-technical and social roles (Iivari & Kerola, 1983; Iivari & Hirschheim, 1996). According to the technical view, the emphasis of IS development lies in technical matters, and the possible social implications of IS development are at best afterthoughts. Social schools emphasize the development of organizational systems (before technical matters); and the sociotechnical view contends that technical and organizational systems are equally important (Iivari & Hirschheim, 1996).

Research approaches

The research approach viewpoint indicates how the development approaches themselves are developed and validated. For example, a question such as "are approaches validated empirically or conceptually?" can be answered by indicating the used research approaches. The classification of research methods is adapted from Järvinen (1997). According to our knowledge, there are other classifications of research approaches and methods including Jenkins (1985); Galliers & Land (1987); Goubil-

Gambrell (1991); Iivari (1991b); Nunamaker et al. (1991); Stohr & Konsynski (1992); March & Smith (1995) and Wynekoop & Russo (1997). The one by Järvinen (1997) was chosen since it is systematic and holistic.

Applicability into IS development

The problem of developmental duality means that the normal system development and security development are separate activities having conflicting requirements among other weakness (Baskerville, 1992). Due to such conflicts, separate security methods (i.e. those that cannot be integrated into normal IS development) should be eliminated altogether (Baskerville, 1993 p. 410). The problem of developmental duality can be retraced to Ockham's razor "Plurality should not be assumed without necessity" (Baskerville, 1988 p. 93). Ockham's razor briefly means: keep it simple. In other words, in the case of compelling theories, for example, the preference should be given to the simplest theory.

The most extensive formulation of Ockham's (1990) razor is also worthwhile to note: "Nulla pluralitas est ponenda nisi per rationem vel experientiam vel auctoritatem illius, qui non potest falli nec errare, potest convinci". This means that no plurality should be accepted unless it can be proved (i) by reason, or (ii) by experience, or (iii) by some infallible authority (Ockham, 1990). Let us apply this version of Ockham's razor/eraser to IS security methods in a pragmatic sense. Without IS it is difficult to see something called IS security. To have ISS, it seems rational that there is something called IS (we, however, do not thereby claim in ontological argument that IS exists in the fundamental ontological sense). If there were no IS, would there be any (pragmatic) need for ISS? IS is a human construction. IS does not rain down from sky - so to have an IS, we need to develop one. So to develop IS we need to use a method (whether the method is formal, semi-formal, or totally informal). Hence, to develop an IS we are likely to adopt a method (remember our loose use of method). Thus, it generally seems that the "existence" of IS requires a method, of which result, the IS is developed. So, the "existence" of an IS method (or development process) is quite necessary. By contrast, an IS exists without any security development - though it would perhaps be insecure (and this may cause certain complications). In that respect, security method/development is, in a pragmatic sense - dependent upon the existence of IS (and IS method, by which the IS was built). In turn, IS security is not a necessary prerequisite for the development of IS (though security may be important for carrying out the operations of IS successfully) - the IS can exist (as a human construction) without security development. In other words, IS security is ontologically dependent (using term by Niiniluoto, 1999 p. 27) on IS (ISS could not exist without IS existing). When separated

security methods are applied to existing IS, the plurality comes into play (as described by Baskerville (1988; 1992)): security development may have its own requirements (that are conflicting with IS development), etc. Therefore, we can conclude that in a general sense the plurality should be avoided and we showed that the source of plurality was the separate ISS method, since generally an IS development method is more a prerequisite for IS than ISS method. Corollary, Ockham's razor insists that stand-alone security method should be eliminated.

Meta-model for IS

The meta-model used is one by Iivari (1989). It is based on a commonly agreed separation between three levels of modeling/abstraction for an IS (Iivari & Koskela, 1987; Iivari, 1989; Lyytinen, 1987): 1. The organizational level, which defines the organizational role and context of the IS. 2; The conceptual level, which defines an implementation-independent specification for the IS.3; The technical level, which defines the technical implementation for the IS. Originally (Iivari, 1989), the levels are in order of abstraction. Hence, for example, the conceptual level can be seen as an abstraction of the organizational level. Since the meta-model is based on the commonly agreed levels of IS, it shows which aspects of information systems are covered by different methods. In other words, it provides a framework for analyzing the breadth of each developmental approach. However, it does not pay attention to the relevance of the content (including processes, notations, etc) of the approaches. For example, the meta-model, per se, does not give much information about such issues as ease of use, tool support, or conflicts within the processes, etc.

3. NORMATIVE STANDARDS

Normative standards include checklists (AFIPS, 1979; Wood et al., 1987; Cooper, 1989; Custance, 1996; Moulton & Moulton, 1996), management standards (e.g. Code of Practice/BS 7799, 1993; CobiT, 1995; GASSP, 1999; IT Protection Manual, 1996) and non-technical evaluation and maturity criteria (see e.g. Chokhani, 1992; Abrams & Podell, 1995).

The evaluation, management and maturity standards can also be included into a category of development methods since they strongly guide development (e.g. by improving processes or products). As they all propose norms for developing or managing secure ISs they can be classified as normative standards (Siponen, 2001b). Many of the evaluation criteria such as TCSEC/Orange Book and Common Criteria are inadequate from an

IS/management/organizational perspective since they focus on technical or implementation level issues.

The security maturity standards such as the System Security Engineering Capability Maturity Model (henceforth SSE-CMM) differs from traditional checklists (and similar standards) in two respects. First, SSE-CMM has a non-organizational/public dimension, as well. This means that SSE-CMM - ideally - shows the security level of organization for partners and customer, for instance (SSE-CMM, 1998b). However, from the viewpoint of the organization, this means that the standard is more seriously adapted to guide development. Secondly, SSE-CMM has a concept of process areas (e.g. see Ferraiolo & Sachs, 1996) that is similar to "second generation mechanistic methods" (classified by Baskerville, 1993) that pays attention to the organizational differences. Other maturity approaches include Stacey's (1996) approach reflecting on CMM and Murine's & Carpenter's (1984) SSM that measures system security using software security metrics, which is based on software quality metrics (SQM). According to our knowledge, they have not gotten common recognition.

Another recent effort to build widely accepted evaluation criteria is the Common Criteria (CC), which is focused on products and processes. The validation of CC is based on expert validation. One of the important difference between SSE-CMM and CC is that CC does not take into account non-technical aspects (Overbeek, 1995).

The research objective

The research objective behind the checklist is means-oriented. The aim is secure information systems by implementing a certain set of solutions.

The organizational role of IS security

The organizational role of Information Systems security is technical. The primary focus of the secure IS developed rests on technical issues. The organizational structures and social implications come as afterthoughts.

The research approaches and the meta-model

According to our knowledge, the normative standards are based on authors' experiences. In that way, we really cannot say that they are developed using a certain research approach. If the authors' observations and results thereof were available, we might be able say that they are based on theory creating or theory testing research (cf. Järvinen, 1997). Checklists/normative standards provide only organizational level support for designing secure IS.

Applicability in the IS development process

Checklists do maintain the dualistic development, meaning that security and normal ISD are developed separately, having conflicting requirements, among other weaknesses (Baskerville, 1988; 1992).

4. RISK MANAGEMENT TECHNIQUES

Risk management (RM) approaches have been traditionally used in the field of IS after its development in the nuclear arena (Tarr & Kinsman, 1996) and is a de facto topic of non-technical textbooks (e.g. Gollman, 1999; Norman, 1983; Parker, 1998). Several RM approaches have been presented (Wong, 1977; Cooper, 1989; Custance, 1996; Veatc et al., 1995; Moses, 1995; Bennett & Kailay, 1992; Halliday et al. 1996; Lichtenstein, 1996; Freeman et al., 1997; Jung et al. 1999; Spruit & Samwel, 1999). The terms risk analysis/management/assessment are used very differently by different authors, and without muddling through the terminological mess, we hereafter apply the term RM.

The research objectives

As for the research objective, RM techniques are both 1) means-end oriented and 2) interpretive. They are clearly means-oriented since the aim of risk management is to provide feasibility justification, as mentioned. There have also been reasons reported for why RM is interpretive. For example, Baskerville (1991b) argues that the role of risk management is interpretive. We understand his view as follows: Baskerville (1991b) seems to acknowledge that RM is inadequate as a means-oriented tool, but that it may be valid as an interpretive method (RM provides clear numbers for managers). Also, Guarro sees that the objective of risk management is interpretive. The aim of RM is to understand the environment (Guarro, 1987).

The organizational role of Information Systems

The organizational roles of IS security are mainly technical. For the RM community (generally speaking), the technical system is the first preference and the issues concerning social systems come second.

The research approach and Meta-model

Risk management approaches are based on conceptual analysis. Risk management provides only organizational level support for designing secure IS.

Applicability in the IS development process

The risk management approaches maintain the problem of developmental duality: There is no explicit guidance about how RM could be integrated to IS or software development process.

5. FORMAL MODEL DEVELOPMENT

Formal model-oriented development (FMD) holds that IS or SW development should be based on formally validated components or carried out by formal methods. Formal refers to use of logic as the reference discipline - preferably hard analytic philosophy (see the philosophical assumptions) - by which the security of the solutions can be validated, i.e. meet certain requirements. This appears to be held by a majority of computer science security researchers: the crucial problem behind insecure systems is the lack, or wrongful use or implementation, of formal development (e.g. O'Leary et al., 1990; Parnas et al. 1990 p. 647; Anderson, 1993; Freeman & Neely, 1993; Williams & Abrams, 1995; Barnes, 1998; McDermott & Fox, 1999).

The research objectives

The research objective is means-oriented - to provide a tool for reliable and secure implementation.

The organizational role of Information Systems

The view of the organizational role of IS is technical. The design objective lies in technical systems. Poor technical quality is behind the security problems, because the most important condition for achieving secure systems is technical quality.

The research approach and Meta-model

As can be seen, the favored research approach is mathematical modeling. All "modeling" support is concentrated on a technical level in terms of Iivari's meta-model (Iivari, 1989). Formal model approach does not propose any organizational or conceptual modeling means.

Applicability into IS Development Process

FMD maintains the duality problem. Suggestions for the integration of security and normal ISD/SW development have been proposed, including Zhou et al. (1999). These are, however, mainly concentrated on implementation (some even more specified) issues, ignoring logical level issues (e.g. modeling). According to Evans & Welling (1999), formal

methods are even rejected by many practitioners because they are regarded to be too lower level. The integration of security development and normal ISD is also difficult due to differences between notation and approaches (Evans & Welling, 1999). Normal IS development is rarely carried out in a formal manner.

6. COMMON SENSE PRINCIPLES

Common sense principles (CSP) refer to principles (i.e. loose guidelines, but not as holistic guidelines as checklists) that are "validated" or reasoned by the authors' own experiences. As many of them are accepted (e.g. Garfinkel & Spafford, 1997; Parker, 1998 p. 329-330; Summers, 1997 p. 250-252; Zurko & Simon, 1996), though these principles are based on a less disciplined development process (not validated in a scientific manner), we may call them as CSP. The difference between the "principles" and checklists and other normative standards is that the "principles" 1) are more abstract than checklists; and 2) are more guiding and not argued to be universally valid, while checklist are argued to be, somewhat universally valid. There are several other or modified CSP's, such as proposed by Fisher (1984), Essinger (1992), Fites & Kratz (1993), Finne (1995), OECD (1996), Sherwood (1996), Coyle et al. (1997), Parker (1998) and Nitzberg (1999). There are no studies testing the principles in practice, for example. As the principles are very abstract and not systematic, they do not per se form a process that could guide development.

The research objective

The CSPs are generally means-oriented. The research objective is to produce guidance, by the help of which the goal, i.e. more secure systems, can be achieved.

The organizational role of ISS

Generally, the organizational roles of IS security of the different principles are technical, though there are exceptions, such as Angel (1993), who seems to hold a social view.

The research approach and Meta-model

We did not find any research approaches that were used to develop the principles. It is possible that empirical research, particularly theory testing and theory creating, as well as conceptual analysis, could be used to develop these principles (and perhaps are used unconsciously, and therefore are not reported). In terms of Iivari's (1989) meta-model for IS, these principles

provide only organizational level (functional abstraction since they may be understood as work procedures) support for security development.

Applicability in IS development

The principles are faced with the problem of developmental duality. They do not propose any means by which these principles can be integrated to normal IS development. Moreover, principles such as separation of duty may often conflict with information systems normal requirements.

7. DISCUSSION AND IMPLICATIONS FOR RESEARCH AND PRACTICE

The implications summarized in Table 2.

Table 2. Implications in the light of different viewpoints.

Viewpoints	Findings	Implications
Research objectives	Mainly means-oriented	Alternative approaches are needed
Organizational role of IS security	Mainly technical	Alternative approaches (more socio-technical, social) are needed
Research approaches	The conceptual analysis was the research approach most used	Additional empirical studies are needed
Applicability	Conventional approaches cannot be integrated into IS or software development	Conventional approaches cannot be integrated into IS development. More guidance is needed about how the approaches could be integrated into IS development
Meta-model for IS	Approaches were not comprehensive: they give only organizational level support	Given that all levels of IS are relevant to the model, new approaches that can provide comprehensive support are needed

These implications will be discussed next.

Research objectives: As for conventional methods, the most commonly held view about the objectives of that IS security research is means-oriented. It is widely suggested that due to the social dimensions of IS, alternative approaches, particularly the interpretive approach, are needed (Klein & Lyytinen, 1985; Hirschheim, 1985; Walsham, 1996; Galliers & Swan, 1997; Klein & Myers, 1999). It is postulated that critical approaches are needed as

well. The development cannot be based on "blind" approaches - the risks are far too high, for our assumptions may be proven wrong in the final analysis. Therefore, critique plays an essential role by keeping us on our toes, and forcing us to prove our ideas.

The most commonly held *organizational role of IS security* was the technical view. This results in practitioners having only technical and a few socio-technical approaches available from which they can choose an IS security development method. Recently, many studies (e.g. Dhillon & Backhouse, 2000; Dhillon & Backhouse, 2001; Baskerville, 1988; Dhillon, 1997) have strongly advocated the relevance of the socio-technical role. They mainly argue that a technical "engineering" approach is too technical in an organization since an organization is a social institution.

Research approaches used. The conceptual analysis/intuition is used to develop all the conventional approaches excluding FMD, which uses mathematical modeling. Disciplined empirical studies from a wide cross-section - 1) in which neither the research process nor the results are secret and 2) all possible variables are considered - as well as real conceptual analysis (which is not based on intuitions and which takes the relevant research and objections into account) are needed.

Applicability into IS/software development process: Conventional approaches cannot be integrated into IS development. Given that the development and use of conventional approaches is still desired, it is suggested that more guidance about the integration of the conventional approaches into IS development is needed.

The meta-model for IS: The conventional ISS approaches only cover the organizational level, except for FMD, which provides support on technical levels, as well. As a result, more holistic approaches that cover all levels of IS (organizational, conceptual, technical) are needed.

8. CONCLUSIONS

Conventional approaches for secure IS development were explicated: checklists/normative standards, common sense principles and formal development. Whese approaches were analyzed from the viewpoints of I) the research objectives; II) the organizational role of IS security; III) research approaches used; IV) applicability; and V) a conceptual meta-model for IS.

The dominating research objective is means-oriented. The research approaches used range from mathematical modeling (formal methods) to conceptual analysis (risk management). Risk management techniques have traditionally been means-end oriented, but recently their interpretive roles have been recognized. Common sense principles and normative are not

based on any research approach/method. The dominating organizational role of IS security is technical (normative standards, risk management, formal methods and most common sense principles).

The conventional approaches are not applicable to IS or software development, resulting the problem of developmental duality.

As for the meta-model for IS, the approaches are within organizational (standards, risk management, common sense principles) and technical contexts (formal development).

9. REFERENCES

- Abadi, M. & Needham, R., (1994), Prudent Engineering Practice for Cryptographic Protocols. Proceedings of the 1994 IEEE Symposium on Research in Security and privacy.
- Abrams, M.D. & Bailey, D., (1995), Abstraction and Refinement of Layered Security Policy. In: Information Security - An integrated Collection of Essays. Edited by M. D. Abrams, S. Jajodia & H. J. Podell. IEEE Computer Society Press, Los Alamitos, California, USA.
- Abrams, M.D., & Podell, H.J., (1995), Evaluation issues. In: Information Security - An Integrated Collection of Essays, Edited by M. D. Abrams, S. Jajodia & H. J. Podell, IEEE Computer Society Press, CA, USA.
- Angel, I., (1993), Computer Security in these uncertain times: the need for a new approach. Proceedings of the 10th International Conference on Computer Security, Audit and Control (CompSec), London, October.
- Anderson, R., (1993), Why Cryptosystems Fail. Communication of the ACM, November, vol. 37, no. 11., pp. 32-44.
- Barnes, B.H., (1998), Computer security research: a British perspective. IEEE Software. Volume 15, no 5, Sept.-Oct. Pp. 30 -33.
- Baskerville, R., (1988), Designing Information Systems Security. John Wiley Information Systems Series.
- Baskerville, R., (1991a), Risk Analysis: An Interpretative Feasibility Tool In Justifying Information Systems Security. European Journal of Information Systems Vol. 1, Issue 2, pp. 121-130.
- Baskerville, R., (1991b), Risk Analysis As A Source of Professional Knowledge. Vol. 10, Issue 8, pp. 749-764.
- Baskerville, R., (1992), The Developmental Duality of Information Systems Security. Journal of Management Systems. Vol. 4, no. 1, pp. 1-12.
- Baskerville, R., (1993), Information Systems Security Design Methods: Implications for Information Systems Development. Computing Surveys 25, (4) December, pp. 375-414.
- Bennett, S. P., & Kailay, M. P., (1992), An application of qualitative risk analysis to computer security for the commercial sector. Proceedings of the Eighth ACM Annual Computer Security Applications conference.
- Blakley, B, Kienze, D.M., (1997), Some Weaknesses of the TCB model. Proceedings of the 1997 IEEE Symposium on Security and Privacy. IEEE Computer Society Press.
- Booyens, H.A.S., & Eloff, J.H.P., (1995), A Methodology for the development of secure Application Systems. In proceeding of the 11th IFIP TC11 international conference on information security, IFIP/SEC'95.
- Chokhani, S., (1992), Trusted products evaluation. Communications of the ACM. Vol. 35, Issue 7, pp. 64-76.

- Chua, W.F., (1986), Radical Developments in Accounting Thought. *Accounting Review*, vol. 61, issue 5, pp. 583-598.
- Code of Practice for Information Security Management, (1993), Department of Trade and Industry. DISC PD003. British Standard Institution, London, UK.
- Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and General Model. May 1998, Version 2.0, CCIB-98-026.
- Cooper, J.A., (1989), *Computer and Communications Security: Strategies for the 1990s*. McGraw-Hill, New York, USA.
- Custance, N.D.E., (1996), The use of baseline measures in risk assessment. *Proceedings of the 30th Annual International Carnahan Conference on Security Technology*. IEEE Computer Society Press.
- Dhillon, G., (1997), *Managing Information Systems Security*. MacMillan Press LTD, UK.
- Dhillon, G. & Backhouse, J., (2000), Information system security management in the new millennium. *Communications of the ACM*, Volume 43, Issue 7, pp. 125-128.
- Dhillon, G. and Backhouse, J., (2001), Current directions in IS security research: toward socio-organizational perspectives. *Information Systems Journal*. Vol 11, No 2.
- Evans, A.S. & Welling, A.J., (1999), UML and the formal development of safety-critical real-time systems. *IEE Colloquium on Applicable Modelling, Verification and Analysis Techniques for Real-Time Systems*.
- Ferraiolo, K., & Sachs, J.E., (1996), Distinguishing Security Engineering Process Areas by Maturity Levels. *Proceedings of the 9th Annual Canadian Information Technology Security Symposium*.
- Finne, T., (1995), The Information Security Chain in a Company. *Computers & Security*. Vol. 15, No. 4, pp. 297-316.
- Fisher, R.P., (1984), *Information Systems Security*. Prentice-Hall, New Jersey, USA.
- Fites, P. & Kratz, M.P.J., (1993), *Information Systems Security: A Practitioner's Reference*. Van Nostrand Reinhold, New York, USA.
- Fitzgerald, K.J., (1995), Information security baselines. *Information Management & Computer Security*, Vol. 3 Issue 2, pp. 8-12.
- Fitzgerald, K.J., (1993), Risk Analysis: Ten Years On. *Information Management & Computer Security*, Vol. 1, issue 5.
- Freeman, J.W. & Neely, R.B., (1993), On security policy modeling. *Proceedings of the eight Annual Conference on Computer Assurance (COMPASS'93)*.
- Freeman, J.W., Darr, T.C., Neely, R.B. (1997), Risk Assessment for large heterogeneous systems. *proceedings of the 13th Annual Computer Security Applications Conference*.
- Galliers, R.D., & Land, F.F., (1987), Choosing appropriate information systems research methodologies. *Communication of the ACM*, vol. 30, no. 11, pp. 900-902.
- Galliers, R.D., & Swan, J.A., (1997), Against structured approaches: information requirements analysis as a socially mediated process. *Proceedings of the Thirtieth Hawaii International Conference on Systems Sciences*, IEEE Society Press.
- Garfinkel S. & Spafford G., (1997), *Web Security & Commerce*. O'Reilly & Associates, Inc. USA.
- Garvey, T.D., (1992), The Inference Problem for Computer Security. *Proceedings of the Fifth Computer Security Foundations Workshop*. IEEE Computer Society Press.
- GASSP, (1999), *Generally Accepted System Security Principles (GASSP)*. Version 2.0. *Information Systems Security*. June, vol. 8, no. 3
- Gollman, D., (1999), *Computer Security*. Wiley & sons, UK.
- Guarro, S.B., (1987), Principles and Procedures of the LRAM Approach to Information Systems Risk Analysis and Management. *Computer & Security*. Issue 6, pp. 493-504.

- Halliday, S. Badenhorst, K., von Solms, R., (1996), A business approach to effective information technology risk analysis and management. *Information Management & Computer Security*, Vol. 4 Issue 1, pp. 19-31.
- Hefner, R., (1997b), A process standard for systems security engineering: development experiences and pilot results. Third IEEE International 1997 Software Engineering Standards Symposium and Forum, Emerging International Standards (ISESS 97).
- Hirschheim, R., (1985), Information systems epistemology: An historical perspective. In: *Research methods in information systems*. E. Mumford et al. (eds), Elsevier Science Publisher.
- Hirschheim, R., Klein, H. K., & Lyytinen, K., (1995), *Information Systems Development and Data Modelling: Conceptual and Philosophical Foundations*. Cambridge University Press, UK.
- Iivari, J. & Kerola, P., (1983), A Sociocybernetic framework for the feature analysis of information systems design methodologies. In T.W. Olle, H.G. Sol, C.J. Tully (eds.), *Information Systems Design Methodologies: A Feature Analysis*. Pp. 87-139, North-Holland, Amsterdam.
- Iivari, J & Koskela, E., (1987), The PICO model for IS design, *MIS Quarterly*, Vol. 11, No. 3, pp. 401-419.
- Iivari, J., (1989), Levels of abstraction as a Conceptual Framework for an Information Systems. In E. D. Falkenberg and P. Lindgreen (eds): *Information System Concepts: An In-depth Analysis*. North-Holland, Amsterdam.
- Iivari, J. and Hirschheim, R., (1996), Analyzing information systems development: A comparison and analysis of eight IS development approaches, *Information Systems Information Technology Security Evaluation Criteria (ITSEC) (1990), Harmonised Criteria of France, Germany, The Netherlands and the United Kingdom*.
- IT Baseline Protection Manual, (1996), BSI, Germany.
- Jackson, F., (1980), Ontological Commitment and Paraphrase. *Philosophy*, Vol. 55, no. 213, pp. 303-315.
- James, H.L., (1996), Managing information systems security: a soft approach. *Proceedings of the Information Systems Conference of New Zealand*. IEEE Society Press.
- Järvinen, P., (1997), The new classification of research approaches. The IFIP Pink Summary - 36 years of IFIP. Edited by H. Zemanek, Laxenburg, IFIP.
- Jung, C., Han, I., & Suh, B., (1999), Risk Analysis for Electronic Commerce Using Case-Based Reasoning. *International Journal of Intelligent systems in Accounting, Finance & Management*. Vol. 8, issue 1, pp. 61-73.
- Kahn, J.J., & Abrams, M.D., (1994), Editorial: why bad things happen to good systems, and what to do about it. *Proceedings of the 10th Annual Computer Security Application Conference*. IEEE Computer Society Press.
- Klein, H., & Lyytinen, K., (1985), The Poverty of Scientism in Information Systems. pp. 131-161. In: *Research methods in information systems*. E. Mumford et al. (eds), Elsevier Science Publisher.
- Klein, H.K., Myers, M.D., (1999), A set of Principles for Conducting and Evaluating Interpretative Field Studies in Information Systems. *MIS Quarterly*, Vo. 23, No. 1, pp. 67-94.
- Lichtenstein, S., (1996), Factors in the selection of a risk assessment method. *Information Management & Computer Security*, Vol. 4 Issue 4, pp. 20-25.
- Lyytinen, K., (1987), A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations. In R. Boland & R. A. Hirschheim (eds): *Critical Issues in Information Systems Research*, John Wiley & Sons, Ltd., pp. 3-41.

- Mathiassen & Munk-Madsen, A., (1986), Formalizations in System Development. Behaviour and Information Technology, Vol. 5, No. 2.
- Mautner, T., (1996), A Dictionary of Philosophy. Blackwell Publishers Ltd, Oxford, UK.
- Mingers, J.C., (1995), Information and Meaning: foundations for an intersubjective account. Information Systems Journal, Vol. 5, no. 4, October, Pp. 285-306.
- Moore, GE., (1903), Principia Ethica, Cambridge, UK.
- Moses, R., (1995), Corporate risk analysis and management strategies. Proceedings of the European Convention on Security and Detection. IEEE Computer Society Press.
- Moulton, R. T., & Moulton, M. E., (1996), Electronic Communications Risk Management: A Checklist for Business Managers. Computer & Security, Vol. 15, No.5.
- Murine, G.E. & Carpenter, C. L., (1984), Measuring Computer System Security Using Software Security Metrics. In Computer Security: A global challenge, J.H. Finch and E.G. Dougall (eds.). Elsevier Science Publisher.
- Nitzberg, S.D., (1999), The Cyber Battlefield: Is This The Setting for the Ultimate World War? Proceedings of Military Communications Conference (MILCOM). Vol. 1. IEEE Computer Society Press.
- Niiniluoto, I. (1999), Critical Scientific Realism. Clarendon Library of Logic and Philosophy, Oxford University Press, Oxford, UK.
- Norman, A.R.D., (1983), Computer Insecurity. Chapman & Hall, NY, USA.
- Nunamaker, J.F., Chen, M., Purdin, T.D.M., (1991), Systems development in information systems research. Journal of Management Information Systems, vol. 7., no. 3., pp. 89-106.
- Ockham, W., (1990), Philosophical Writings: A selection. Hackett Publishing Company, Indianapolis, USA.
- OECD, (1996), Guidelines for the Security of Information Systems. OECD, Paris, France.
- O'Leary, T.J., Goul, M., Moffitt, K.E. & Radwan, A.E., (1990), Validating expert systems. IEEE Expert, Vol. 5, Issue 3, pp. 51-58.
- Overbeek, P.L., (1995), Common Criteria for IT Security Evaluation - Update Report. Proceedings of the IFIP TC11 Eleventh International Conference on Information Security, IFIP/SEC'95.
- Ozier, W., (1999), Risk Analysis and Risk Assessment. Handbook of Information Security Management (eds): M. Krause and H.F. Tipton, CRC Press LLC, Florida.
- Pap, A., (1949), Elements of Analytic Philosophy.
- Parker, D. B., (1998), Fighting Computer Crime - A New Framework for Protecting Information. Wiley Computer Publishing. USA.
- Parnas, D.L. Schouwen, J. & Kwan, S.P., (1990), Evaluation of Safety-Critical Software. Communications of the ACM, Vol. 33, No. 6, June, pp. 636-648.
- Payne, C.N., Froscher, J.N., McDermott, J.P., (1990), On models for a trusted application system. Proceedings of the Sixth Annual Computer Security Applications Conference.
- Schaefer, M., (1989), Symbol security condition considered harmful. Proceedings of 1989 IEEE Symposium on Security and Privacy.
- Seager, M. Guaspari, D., Stillerman, M & Marceau, C., (1995), Formal Methods in THETA kernel. Proceedings of the 1995 IEEE Symposium on Security and Privacy.
- Sherwood, J., (1996), SALSA: A Method for Developing Enterprise Security Architecture and Strategy. Computers & Security. Vol. 15, no. 6, pp. 501-506.
- Siponen, M.T., (2001a), An analysis of the recent IS security development approaches: descriptive and prescriptive implications. In: G. Dhillon (eds.) Information Security Management - Global Challenges in the Next Millennium, Idea Group (2001).
- Siponen, M.T. (2001b): On the scientific background of information security management standards: a critique and an agenda for further development. The Second Annual Systems Security Engineering Conference), 28 February – 2 March, Orlando, Florida, USA.

- Solms, R., (1997), Can Security Baseline replace Risk Analysis? Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC'97), 14-16 May, Copenhagen, Denmark.
- Solms, R., (1998), Information security management (3): the Code of Practice for Information Security Management (BS 7799). *Information Management & Computer Security*. Vol. 6, Issue 5, pp. 224-225.
- Solms, R., (1999), Information security management: why standards are important. *Information Management and Computer Security*, Vol. 7, Issue 1, pp. 50-58.
- Spruit, M. & Samwel, P.H., (1999), Risk analysis on Internet connection. Proceedings of the IFIP TC11 WG11.2/WG11.2 Seventh Annual Working Conference on Information Management & Small Systems Security.
- SSE-CMM, (1998a), The Model. v2.0. <http://www.sse-cmm.org>.
- SSE-CMM, (1998b), The Appraisal Method. v2.0. <http://www.sse-cmm.org>.
- Stacey, T.R., (1996), Information Security Program Maturity Grid. *Information Systems Security*. Vol. 5, No.2.
- Thomas, R.K. & Sandhu. R.S. (1994). Conceptual Foundations for a Model of Task-based Authorizations. Proceedings of the 7th IEEE Computer Security Foundations Workshop.
- Walsham, G., (1996), The Emergence of Interpretivism in IS research. *Information Systems Research*, Vol. 6, No. 4, pp. 376-394.
- Veatch, J.D., James, J.W., Bosma, P.H., May, T.T., Garner, D.W., Priem, R.G., (1995), Requirements Driven Methodology for conducting risk analyses on unclassified networks. Proceedings of the 29th Annual International Carnahan Conference on Security Technology.
- Williams, J.G. & Abrams, M.D., (1995), Formal methods and models. In: *Information Security - An integrated Collection of Essays*. Edited by M. D. Abrams, S. Jajodia & H. J. Podell. IEEE Computer Society Press, Los Alamitos, California, USA.
- Winograd, T. & Flores, F., (1986), *Understanding Computers and Cognition*. Addison Wesley Publishing Company, USA.
- Wong, K.K., (1977), *Risk analysis and control: a guide for DP managers*. NCC Publications, Southampton, UK.
- Wood, C.C., Banks, W.W., Guarro, S.B., Garcia, A.A., Hampel, V.E., Sartorio, H.P., (1987), *Computer Security: A Comprehensive controls Checklist*. John Wiley & Sons.
- Zhou, D., Kuo, J.C., Older, S., Chin, S. K., (1999), Formal development of secure email. Proceeding of the 32nd Annual Hawaii International Conference on Systems Sciences.

Redefining Information Systems Security: Viable Information Systems

Maria Karyda, Spyros Kokolakis, Evangelos Kiountouzis

Maria Karyda, Evangelos Kiountouzis: Department of Informatics, Athens University of Economics and Business,

*76 Patission Street, Athens GR-10434 Greece, tel. +301-8203555,
fax: +301-8237369, email: {mka, eak}@aueb.gr*

Spyros Kokolakis: Department of Information & Communication Systems, University of the Aegean,

*Samos GR-83200 Greece, tel. +30-273-82001, fax: +30-273-82009,
email: sak@aegean.gr*

Key words: Information Systems, Information Systems Security, Cybernetics, Viable Information Systems

Abstract: Research on Information Security has been based on a well-established definition of the subject. Consequently, it has delivered a plethora of methods, techniques, mechanisms and tools to protect the so-called security attributes (i.e. availability, confidentiality and integrity) of information. However, modern Information Systems (IS) appear rather vulnerable and people show mistrust on their ability to deliver the services expected. This phenomenon leads us to the conclusion that information security does not necessarily equal IS security. In this paper, we argue that IS security, contrary to information security, remains a confusing term and a neglected research area. We attempt to clarify the meaning and aims of IS security and propose a framework for building secure information systems, or as we suggest them to be called, viable information systems.

1. INTRODUCTION

Research on Information Security has evolved on the basis of a well-established theoretical foundation, the essence of which being the commonly accepted definition of Information Security as the preservation of the so-called security attributes of Information, referring mainly to Confidentiality, Integrity, and Availability. Consequently, research on Information Security has produced significant results, which are rapidly turning into commercial products.

However, a number of security surveys show that Information Systems (IS) suffer severely from security breaches and even the most sophisticated systems appear to be vulnerable to well-coordinated attacks (see for example [CSI, 2000; Ernst&Young, 2000]). The above paradox reveals the significant gap keeping apart IS security from information security.

Contrary to Information Security, IS security lacks a widely accepted definition or at least a common understanding of the meaning and aims of IS security. Therefore, current research on the issue seems fragmented and difficult to be exploited by industry.

The attempt to apply the concept of "security attributes" to the area of IS has little chance of providing an adequate conceptual basis for research and practice. An information system cannot be simply defined as a system that processes data and delivers information. An IS comprises hardware, software, data, procedures and, above all, people. The above elements are in constant interaction and interdependence, forming a complex and dynamic whole. IS belong to a special category of systems usually referred to by the term "human activity systems" [Checkland and Holwell, 1998]. In our view,

an information system is a human activity system comprising five elements, namely hardware, software, data, procedures, and people, interacting with each other and with the environment, aiming to produce and handle information, in order to support human activities in the context of an organisation.

In this perspective, the content and goals of IS security need further elucidation. In the rest of this paper we shall attempt to address the following issues:

- How do we perceive the meaning and aims of IS security?
- How can we build secure information systems?

2. PREVIOUS RESEARCH

The goal of IS security has traditionally been the protection of the three basic information security attributes, confidentiality, availability and integrity, along with some others, such as authentication, privacy, and non-repudiation. Often, security goals would be extended to include also the protection of the information technology infrastructure, such as workstations, servers, and communication lines. This can be achieved in a systematic and well-documented way, using for example the *risk analysis* methodology [Baskerville, 1991]. This systematic view, employed by many of the models, methodologies, techniques and tools, emphasize the protection of the technical components of an IS. As a result, security problems associated with the human factor, as well as managerial and social security problems have been either neglected or treated as technical ones.

Moreover, previous research in IS security stresses also the fact that "...while security traditionally has been focused on confidentiality of information, the problems of greatest concern today relate to the availability of information and continuity of services..." [Lipson and Fisher, 1999]. Many researchers criticize as well the view of security as the preservation of confidentiality, integrity and availability as "dangerously oversimplified" [Parker, 1996] and emphasize the need for addressing security at an "overall level" [Ellof and von Solms, 2000]. The need for a distributed and more flexible IS security management has also been recognized as a necessity, in contrast with the current rigid and centralized type of security management applied in most organizations [Baskerville, 1997].

The obvious shortcomings of the use of the systematic approach described above are addressed in methodologies that apply a systemic view. These methodologies, such as the *Virtual Methodology* [Hitchings, 1996] and *SIM-ETHICS* [Warren, 1996], include human and contextual issues as well as technical solutions and emphasize on the analysis of the organization and relevant systems. The systemic view has also been applied to IS security education, in the holistic approach proposed by Yngstrom [1996].

The dependence of organizations on their IS to maintain their functionality stresses furthermore the importance of the unhindered function of the IS. To address this need, a new approach has been recently introduced focusing on the *survivability* of the IS, with survivability meaning "...the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures or accidents..." [Ellison et al., 1999]. The aim of this new trend is not only to thwart possible intruders or prevent accidents in the premises of the IS, but also to ensure that the required services are delivered, despite the occurrence of unwanted events [Lipson and Fisher, 1999].

The survivability approach emphasizes the importance of the protection of mission-critical systems, using a risk-management perspective that requires the participation of the organization. This approach, however, despite that it offers a very useful view of security, it is narrowly focused on risk-mitigation strategies and contingency planning concepts.

We argue that an IS should not only be considered in terms of its “own survivability”, but in relation to the organization it serves. We therefore use instead the term “viability”, as used in the field of organizational management, according to the *Viable System Model* proposed by Beer [1979; 1981]. In this paper we propose a methodology for building a *viable information system*, which not only retains its capability of offering the required services under different circumstances, but also it functions in the context of the organization, in terms of goal achievement and cost. In our view, this methodology extends the “survivability” approach, by using a systemic model that addresses both the problem of dealing with unwanted events, which threaten the system’s functionality or performance, and the issue of selecting and implementing the appropriate countermeasures so as to achieve “viability”.

3. SYSTEMS VIABILITY AND INFORMATION SYSTEMS SECURITY

Nowadays organizations depend heavily on their IS not only for their functions and operations on a daily basis, but also as a key organizational component in their strategic plans. Furthermore, new organizational forms, which rely almost entirely on their information technology infrastructure and their information systems, have already been established, usually referred to as the Virtual Corporation, Network Organization, or Virtual Organization [Davidow and Malone, 1992; Mowshowitz, 1997].

In general, most of the problems and challenges organizations and IS face today, are more or less similar: both the organization and the IS have to deal with their complexity and manage unexpected changes that occur in an accelerating rate. In addition to this, the effort to overcome these problems is obstructed by the interdependencies between their parts or subsystems. In order for organizations and IS to face the previously mentioned challenges in an effective way, these systems should at least:

- Be able to meet the demands and changes of the environment;
- Have internal structures that can deal with the demand for learning and for quick adaptation; and

- Have communication abilities for connecting and transmitting information

IS operate within the context of the organization they serve, so they can be considered as an organizational function that embraces information technology, information activities (roles, tasks and functions) and organizational activities. We can furthermore refine the IS function as follows [Jayaratha, 1994]:

- i) Information processing and usability function.
- ii) Education and learning function.
- iii) Information systems development function.
- iv) Management and control function.
- v) Strategy and planning function.

Within this functional point of view, it is very hard to distinguish exactly between the IS and the organization it serves. Thus, it is easy to understand why threats to an IS and their impact concern in such a high degree the organization. However, the IS remains the serving system, whose functionality needs to be protected and preserved, in order for the served system, the organization, to maintain its existence within its environment.

Ashby [1964] argued that only variety can control variety (Law of Requisite Variety). By this he meant that if a situation was complex, with many variables, then the techniques for dealing with the situation would need to have the same amount and kind of variety. If Ashby's Law of Requisite Variety is accepted this means that the risk analysis techniques used to establish security measures must have at least the same kind and level of knowledge as the intruders themselves. However, while organizations change, technology changes, plain risk analysis techniques, usually based on software packages, i.e. CRAMM, remain unchanged, or, at least, change with a small rate (time lag). In other words, risk analysis techniques are static.

On the other hand, it is evident that there is consensus among many that the use of methodologies is positive and well advised. However, practitioners have been somewhat slow in adopting IS security methodologies. This could be explained variously as, for example, due to the ignorance syndrome among the designers, or the slow speed of technology transfer. However, although methodologies are attractive and have an intuitive appeal, the fact is that the methodology is merely a framework for organizing the process.

Moreover, IS security is a managerial problem and therefore should not be addressed as a separate problem, instead IS security management should be

incorporated into organization management and should change with it. This means that IS security should be a build-on characteristic and not an add-on one.

In our view, IS security should preserve the ability of the IS to deliver the required services to the organization, but most important to achieve the most effective coupling between the IS and the organization. The goal of IS security should be the protection of the functionality of the IS, not necessarily of the IS itself or its components, provided that the IS achieves the goals, which have been established by the organization, and operates within a certain scope.

A system that is able to maintain an independent existence in the long run and within a dynamic environment is called a *viable* system. In this paper, we redefine the issue of designing *secure information systems* by designing *viable information systems*. According to this approach, a viable information system is capable of maintaining its existence by managing the risk that stems either from inside or the environment.

3.1 The Viable System Model

As one of the basic tools in our approach we use the Viable Systems Model (VSM) as proposed by Stafford Beer in the early 1970s. VSM is the outcome of Beer's thirty-year effort to elucidate the laws of management, by combining his expertise in cybernetics and his study of biological systems. Beer found that all organisms displaying viability (viability being the capability to maintain an independent existence in the long term) share five basic properties [Brocklesby and Cummings, 1996]. These properties are “*five necessary and sufficient subsystems interactively involved in any organism or organization that is capable of maintaining its identity independently of other such organisms within a shared environment.*” [Beer, 1984] Beer also explains that this ‘set of rules’ has not been created by way of analogy between an organism and an organization, but the rules were “*developed to account for viability in any survival-worthy system at all*” [Beer, 1984].

In brief, these systemic functions are:

- *System One.* The ‘operational elements’ that produce the system and interact with the external environment. These elements are themselves viable systems.
- *System Two.* The ‘co-ordination’ functions that ensure that the operational elements work harmoniously.

- *System Three.* The ‘control’ activities, which maintain and allocate resources to the operational elements.
- *System Four.* The ‘intelligence’ functions that consider the system as a whole -its strategic opportunities, threats and future direction. They also interface with the environment.
- *System Five.* The ‘identity’ function, which identifies self-awareness in the system.

3.2 Viable Information Systems

We have already described the need to address security needs within information systems in a holistic and systemic way, arguing that attempts to introduce the well-founded concept of information security in the information systems field have not been fruitful. Our aim is to build a viable information system, rather than a secure one. A viable information system possesses the ability to maintain its existence, by managing risk and, hence, we can apply the Viable System Model (VSM) as proposed by Beer.

4. BUILDING A VIABLE INFORMATION SYSTEM

We propose a three-phase iterative process for building a viable information system, namely **diagnosis**, **re-design**, and **transformation** (see Figure 1).

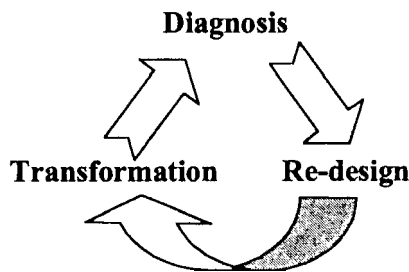


Figure 1. Three phases for building viable information systems.

4.1 Diagnosis

We call the first phase Diagnosis, since it is the phase at which one has to detect vulnerabilities, defects and other factors that threaten the system’s

viability. This will determine the kind of intervention needed to resolve these problems. We use VSM for this task, since it is an effective and powerful tool for detecting inefficiencies and defects within a system, as well as for planning and implementing change. However, before addressing the issue of how to transform an IS into a viable system, one has to assess the IS, by evaluating its contribution to the achievement of the organizational goals. We suggest that three parameters should be considered, i.e. *performance*, *risk*, and *cost*.

4.1.1 4.1.1 Parameter evaluation

System performance refers to the degree the system achieves its goals. It is a measure of the system's contribution to the goals of the organization. If we consider, for example, a production system, the volume of the output it produces can measure the performance.

In real-life systems, performance is never guaranteed and there is always some risk involved. It is, therefore, unrealistic to evaluate a system by its regular performance and not to take into account the possibility of a breakdown. On the contrary, researchers have indicated the need to design IS that "anticipate breakdown" [Winograd and Flores, 1986]. Therefore, we argue that *risk* should also be evaluated. Risk expresses the possibility of a system failing to meet its goal in the future. Finally, a realistic assessment of a system should not overlook *cost*, i.e. the resources used in order to achieve the goals of the system and to mitigate risk.

Similar evaluation methods are quite common in areas such as finance management, where candidate investments are evaluated in terms of anticipated profit, investment cost and risk. However, the application of such methods in the area of IS is not straightforward. Such an evaluation requires a thorough analysis of the IS. For this purpose we use process modeling, which offers a rich model of the IS in the context of the organization it serves. The process modeling technique used in the following example is based on IDEF \emptyset , a popular modeling technique used in business process re-engineering [Mayer et al., 1995]. IDEF \emptyset uses five basic elements: process, input, output, control and mechanism (see Figure 2).

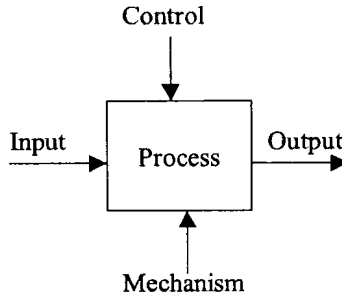


Figure 2. IDEF0 Diagram

In Figure 3, we present the VAT (Value Added Tax) Collection Process, which is part of the Internal Revenue Information System. It should be noticed that this is actually a business process model with a focus on the informational aspects of the process. This is in accordance with our previous argument that in modern organizations IS should not be studied separately from the organizational processes they support.

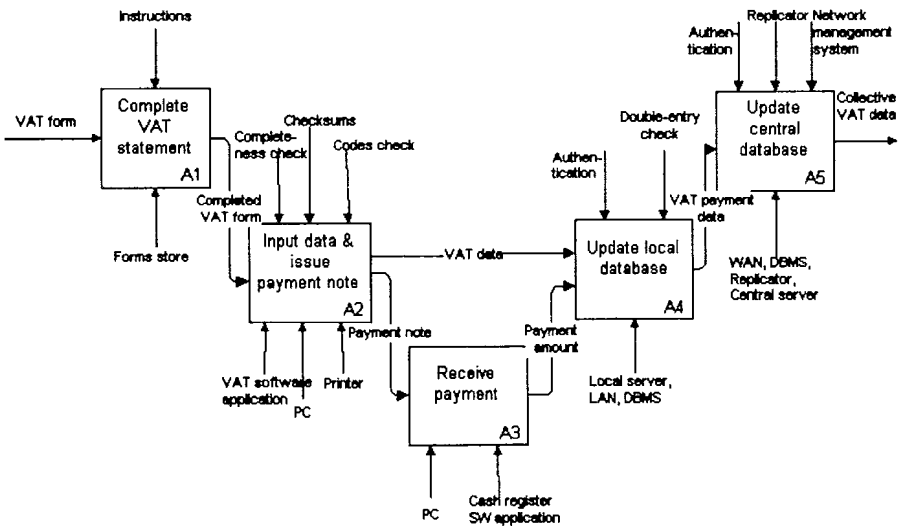


Figure 3. VAT Collection Process

The goals of the VAT Collection Process are: (a) to minimize the time needed to process a VAT statement, (b) to collect the full amount owned by the tax-payer and (c) to protect the privacy of the tax-payer. In the example presented here, the performance of the VAT Collection process is estimated

at an average of 10 VAT statements per hour, with 100% accuracy and 100% success in preserving the confidentiality of personal information given by the tax-payer. Of course, this is the ideal situation; unfortunately the system does not operate as designed all the time.

In order to estimate the level of risk, it is required to identify threats and vulnerabilities in each sub-process and then estimate the total risk level for the VAT Collection Process. It is beyond the scope of the paper to indicate the method to estimate risk, since risk analysis is a well-studied area. The assignment of a risk level in every sub-process forms a "Risk Estimation Diagram" on which we estimate the total risk level for the VAT Collection Process (see Figures 4, 5 and 6). In this case we estimate risk for each of the three goals of the system. In Figure we present a " Risk Estimation Diagram " where a risk factor of 5 (in a 1-100 scale) is estimated, which means that we are only 95/100 confident that the process will achieve its goal.

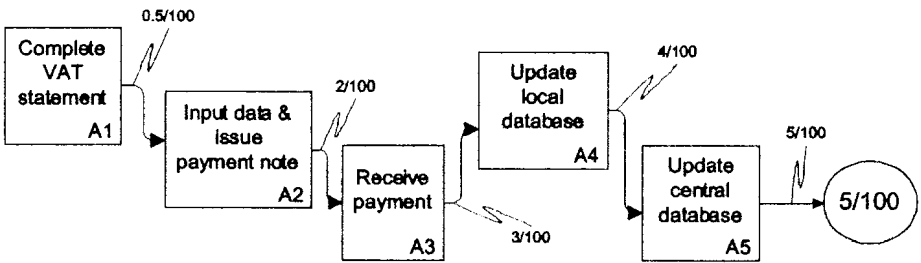


Figure 4. Risk Estimation Diagram for Goal "minimize time needed to process VAT statements"

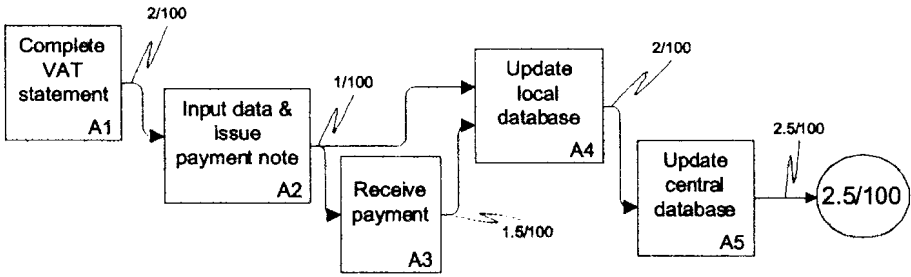


Figure 5. Risk Estimation Diagram for Goal "collect the full amount owned"

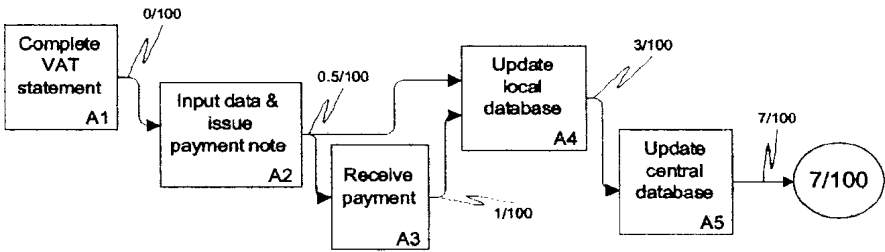


Figure 6. Risk Estimation Diagram for Goal "preserve tax-payers privacy"

In the above figures, we may notice that not all processes increase the level of risk, some processes mitigate risk. For example, A2 in Figure 5 includes several checks that minimize the risk of receiving a false VAT statement.

The last element missing is the estimation of the operation cost. In the case of the VAT Collection Process, cost has been estimated to be 20.00 Euro per hour. The above example is limited to a single process. In order to have a complete model, all processes should be considered and the total Performance, Risk and Cost for the system should be estimated.

4.1.2 4.1.2 VSM analysis

Based on the evaluation of the system, we may improve its current operation by decreasing risk in the processes with a high risk factor (e.g. by including more controls, or adding more resources). However, by this systematic approach we may only achieve minor improvements. Transforming the IS into a viable system requires a more radical approach.

At this point, we suggest the use of VSM as a diagnostic tool. According to VSM a viable system comprises five specific systemic functions (see Section

2). As a first step we should check whether these functions have been adequately developed in the system under study and how they perform. This may lead to designing new processes that implement the missing, underdeveloped or flawed functions.

At the next step we apply VSM techniques to control variety. Variety control provides us with a means to decrease the threats faced by the system. To do this we use the relevant mechanisms applied in VSM, namely the *attenuator*, that can be used to reduce the possible effect of a threat on the system, and the *amplifier*, that enforces the defense of the system.

4.2 Re-design and transformation

Following diagnosis, the IS should be redesigned. The redesign process may include the following steps:

1. Design processes that implement the missing, underdeveloped or flawed VSM functions.
2. Add processes that serve as attenuators or amplifiers.
3. Add controls and mechanisms to mitigate risk for the processes with a high risk factor.
4. Re-evaluate.

The first three steps should achieve the aim of minimizing risk. However, this may result in degrading the overall performance of the system, or increasing the cost. Therefore, re-evaluation is needed, in order to ensure that the proposed changes will really improve the current status of the system.

Finally, when re-design is completed and the proposed changes are approved, the changes should be implemented, in order for the IS to acquire the attributes of a viable system.

5. SUMMARY AND FURTHER RESEARCH

In this paper, we address the issue of building a secure information system. The term IS Security is usually used to refer to the protection of the security attributes of an IS, which, in our opinion, is a very limited way to view the issue. We argue that the term *viable information system* expresses more adequately the concept of the IS which is capable of dealing effectively with threats and contingencies. Furthermore, we suggest that the process of

building a viable information system should follow three phases, namely *diagnosis*, *re-design*, and *transformation*.

The paper, also, contributes a technique for the evaluation of information systems. The proposed evaluation technique considers three parameters, namely *performance*, *risk*, and *cost*. Finally, we show the use of the Viable System Model in building viable information systems.

Further research, may elaborate on the IS evaluation technique and provide a formal specification of it. Moreover, the process-oriented risk modeling diagrammatic technique presented in Section 4 requires further elaboration so as to become an integral part of business (and IS) process modeling.

6. REFERENCES

- Ashby, R.W. (1964). *An introduction to cybernetics*. Chapman and Hall, London.
- Baskerville, R. (1991). Risk analysis: an interpretive feasibility tool in justifying information systems security. *European Journal of Information Systems*, **1**(2), pp.121-130.
- Baskerville, R. (1997). New organisational forms for information security management. In Proceedings of the IFIP/TC11 13th International Conference on Information Security, May 1997, Copenhagen.
- Beer, S. (1984). The Viable System Model: its provenance, development, methodology and pathology. *Journal of Operational Research Society*, **35**, pp.7-26.
- Beer, S. (1979). *The heart of the enterprise*. John Wiley, Chichester, England.
- Beer, S. (1981). *Brain of the firm*. (2nd Edition) John Wiley, Chichester, England.
- Brocklesby, J. and Cummings, S. (1996). Designing a viable organization. *Long Range Planning*, **29**(1), Elsevier Science Ltd.
- Checkland, P. and Holwell, S. (1998). *Information, systems and information systems*. John Wiley and Sons, Chichester, England.
- CSI – Computer Security Institute (2000). *Issues and Trends: 2000 CSI/FBI Computer Crime and Security Survey*. Computer Security Institute, USA.

- Davidow, W. and Malone, M. (1992). *The virtual corporation*, Harper Business, New York.
- Ellison, R., Fisher, D., Linger, R., Lipson, H., Longstaff, T. and Mead, N. (1999). Survivable systems: an emerging discipline. In the proceedings of the 11th Canadian Information Technology Security Symposium (CITSS), Canada.
- Ellof, M. and von Solms, S. (2000). Information security: process evaluation and product evaluation. In the proceedings of IFIP/TC11, 16th Annual Working Conference on Information Security, August 2000, China.
- Ernst&Young (2000). 2nd Annual Global Information Security Survey. Ernst&Young LLP, USA.
- Hitchings, J. (1996). Achieving an integrated design: the way forward for information security. In Ellof, J. and von Solms, S. (eds), *Information Security – the Next Decade*, IFIP SEC'95, Chapman & Hall, London.
- Jayaratha, N. (1994). *Understanding and evaluating methodologies: NIMSAD, a systemic framework*. Mc Graw-Hill, London
- Lipson, H. and Fisher, D. (1999). Survivability – a new technical and business perspective on security. In the proceedings of the New Security Paradigm Workshop June 1999, Canada.
- Mayer, R.J., Benjamin, P.C., Caraway, B.E. and Painter, M.K. (1995). A framework and a suite of methods for business process reengineering. In Grover, V. and Kettinger, W.J. (eds), *Business process change: concepts, methods and technologies*. IDEA Group Publishing, Harrisburg, USA.
- Mowshowitz, A. (1997). Virtual organization. *Communications of the ACM*, **40**(9), pp. 30-37 .
- Parker, D. (1996). A new framework for information security to avoid information anarchy. In Ellof, J. and von Solms, S. (eds.), *Information Security – the Next Decade*, IFIP SEC'95, Chapman & Hall, London.
- Warren, M. (1996). *A security advisory system for healthcare environments*. PhD Thesis, University of Plymouth. U.K.
- Winograd, T. and Flores, F. (1986). *Understanding computers and cognition: a new foundation for design*. Addison-Wesley, USA.

Yngstrom, L. (1996). A holistic approach to IT security. In Ellof, J. and von Solms, S. (eds.), *Information Security – the Next Decade*, IFIP SEC'95, Chapman & Hall, London.

This Page Intentionally Left Blank

EXTENDED DESCRIPTION TECHNIQUES FOR SECURITY ENGINEERING*

Guido Wimmel and Alexander Wisspeintner

Institut für Informatik, Technische Universität München, D-80290 München, Germany

wimmel@in.tum.de, wisspein@in.tum.de

Abstract There is a strong demand for techniques to aid development and modelling of security critical systems. Based on general security evaluation criteria, we show how to extend the system structure diagrams of the CASE tool *AUTOFOCUS* (which are related to UML-RT collaboration diagrams) to allow modelling of security critical systems, in particular concerning components and channels. Both high-level and low-level models of systems are supported, and the notion of security patterns is introduced to provide generic solutions for security requirements. We explain our approach on the example of an electronic purse card system.

Keywords: Security Engineering, Graphical Description Techniques, Software Engineering, Requirements Engineering, Security Properties, Design Patterns, Security Patterns, Formal Methods, CASE, AutoFocus, UML-RT.

1. INTRODUCTION

In developing distributed systems—in particular applications that communicate over open networks like the Internet—security is an extremely important issue. Many customers are reluctant to take part in electronic business, confirmed by recent attacks on well-known portal sites or cases of credit card fraud via the Internet. To overcome their reluctance and make E-Commerce and mobile systems a success, these systems need to become considerably more trustworthy.

To solve this problem, on the one hand there are highly sophisticated collections of evaluation criteria that security critical systems have to meet, like the ITSEC security evaluation criteria (ITSEC, 1990) or their recent successor, the Common Criteria (CC) (Common Criteria, 1999). The Common Criteria

* This work was supported by the German Ministry of Economics within the FairPay project

describe security related functionality to be included into a system, like authentication, secrecy or auditing, and evaluation assurance levels (EALs) for its development. The strictest level is EAL7 (Common Criteria, 1999, part 3, p. 66), where a formal representation of the high-level design is required.

On the other hand, research has produced many formal methods to describe and verify properties of security critical systems, ranging from protocol modelling and verification (Burrows et al., 1989; Lowe, 1996; Paulson, 1998; Thayer et al., 1998) to models for access control, like the Bell-LaPadula model (Bell and LaPadula, 1973) or the notion of non-interference (Goguen and Meseguer, 1998).

Such formal methods however are rarely used in practice, because they require expert knowledge and are costly and time-consuming. Therefore, an integrated software development process for security critical systems is needed, supported by CASE tools and using graphical description techniques. This reduces cost, as security problems are discovered early in the development process when it is still inexpensive to deal with them, and proof of meeting evaluation criteria is a byproduct of software development. In addition, systems developed along a certain integrated “security engineering process” will be much more trustworthy

In this paper, we describe a first step towards using extended description techniques for security modelling. As a basis for our work, we use the AUTOFOCUS description techniques. The AUTOFOCUS (Huber et al., 1998b; Slotosch, 1998; Broy and Slotosch, 1999) system structure diagrams are related to UML-RT collaboration diagrams and describe a system as a set of communicating components. The corresponding CASE tool developed at Munich University of Technology supports user-friendly graphical system design and incorporates simulation, code and test case generation and formal verification of correctness. The main advantage of the use of AUTOFOCUS over a more general description technique as UML is its simplicity. Besides, there exists a clear semantics for the description techniques (for general UML description techniques, defining a formal semantics is still subject of ongoing research). As a start, in our work we focus on security properties of communication channels. We show how certain important security properties of communication channels, such as secrecy and authenticity, can be modelled at different abstraction levels of the system design and explain our ideas on the transition between these levels, using generic security patterns. We give definitions of the meanings of our extended description techniques, based on the AUTOFOCUS semantics. See also (Jürjens, 2001) for first work on integrating access control models into UML description techniques, and (Lotz, 2000) for formal definitions of security properties using the Focus method.

This paper is structured as follows. In Section 2, we give a short introduction to AUTOFOCUS. In Section 3, we present the extensions of AUTOFOCUS

system structure diagrams to model security properties of channels. The usage of these techniques is demonstrated in Section 4, with the help of an example model of an electronic purse system. We conclude in Section 5 with a summary and indicate further work.

2. AUTOFOCUS

AUTOFOCUS/Quest (Huber et al., 1998a; Slotosch, 1998; Philipps and Slotosch, 1999) is a CASE tool recently developed at Munich University of Technology with the goal to combine user-friendly graphical system design and support of simulation, code generation and formal verification of correctness.

AUTOFOCUS supports system specification in a hierarchical, view-oriented way, an approach that is well established and facilitates its use in an industrial environment. However, it is also based on the well-founded formal background Focus (Broy et al., 1992), and a fairly elementary underlying concept: communicating extended Mealy machines.

System specifications in AUTOFOCUS make use of the following views:

- **System Structure Diagrams (SSDs)** are similar to collaboration diagrams and describe structure and interfaces of a system. In the SSD view, the system consists of a number of communicating components, which have input and output ports to allow for sending and receiving messages of a particular data type. The ports can be connected via channels, making it possible for the components to exchange data. SSDs can be hierarchical, i.e. a component belonging to an SSD can have a sub-structure that is defined by an SSD itself. Besides, the components in an SSD can be associated with local variables.
- **Data Type Definitions (DTDs)** define the data types used in the model, with the functional language Quest (Philipps and Slotosch, 1999). In addition to basic types as integer, user-defined hierarchical data types are offered that are very similar to those used in functional programming languages like Gofer (Jones, 1993) or Haskell (Thompson, 1999).
- **State Transition Diagrams (STDs)** represent extended finite automata and are used to describe the behaviour of a component in an SSD. Transitions consist of input patterns on the channels, preconditions, output patterns, and actions setting local variables when the transition is executed. As the main focus of this paper is extending SSDs, we will not describe STDs in detail at this place.
- **Extended Event Traces (EETs)** finally make it possible to describe exemplary system runs, similar to MSCs (ITU, 1996).

The Quest extensions (Slotosch, 1998) to AUTOFOCUS offer various connections of AUTOFOCUS to programming languages and formal verification tools, such as Java code generation, model checking using SMV, bounded model checking and test case generation (Wimmel et al., 2000).

3. EXTENDING SYSTEM STRUCTURE DIAGRAMS

The main difference between security critical systems and traditional systems is the consideration of attacks. A potential third party could try to overhear and manipulate security critical data. To decrease the risk of attacks to a minimum, special security functionalities are used. For example, encryption is a common principle to inhibit overhearing of the communication between two agents.

It is state of the art to use graphical description techniques for system specification. For the specification of security critical systems, we need special description techniques to deal with the particularities of those systems. We extend the AUTOFOCUS description techniques to fulfill the needs of security engineering. In this paper the extensions of the AUTOFOCUS SSDs, mentioned in Section 2, are described. The extensions of the structure diagrams allow the definition of security requirements. Furthermore it is possible to specify the usage of security functionality to fulfill the defined requirements.

We use special tags for the security extensions to the SSDs. These security tags are assigned to particular diagram parts and have a defined semantics. The following sections describe the different security extensions made to the SSDs.

3.1. SECURITY CRITICAL SYSTEM PARTS

To model and evaluate security aspects of distributed systems, it is always necessary to define its security critical parts. The identification of security critical parts should be done very early within the system development process. This task is typically part of the analysis phase. In the Common Criteria, the security critical parts of a system together form the Target Of Evaluation (TOE). The following definition will make our notion of security criticality more precise.

Definition 1 (Security Critical). By security critical parts of a distributed system we mean parts that deal with data or information that has to be protected against unauthorized operations (e.g. disclosure, manipulation, prevention of access etc.). In particular, security critical system parts are connected with security requirements such as secrecy, authentication or auditing, and according to required strictness of evaluation might be subject to formal modelling.

We want to make the distinction visible within the graphical system description. Therefore we annotate security critical system parts with the security tag <<critical>>. Both components and channels can be tagged. To mark non criti-

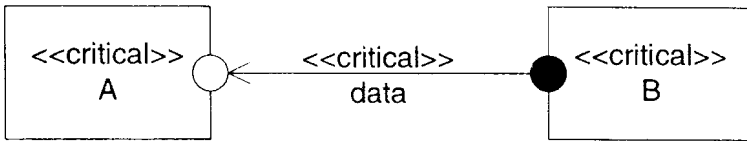


Figure 1 Security Critical System Parts

cal system parts, `<<noncritical>>` is used. A system part without a `<<critical>>` or `<<noncritical>>` tag is non critical by default. Figure 1 shows an SSD. It consists of two security critical components and one security critical channel.

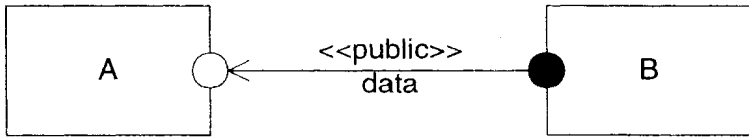
3.2. PUBLIC CHANNELS

The special aspect of security critical systems is the possibility of attacks. Hostile subjects can manipulate the system by overhearing and manipulating the communication between the subsystems. To distinguish private communication channels of a system from public channels, we use the two security tags `<<private>>` and `<<public>>`. A `<<private>>` channel is a channel a hostile party has no access to. The hostile subject can neither overhear nor manipulate the communication that takes place via the channel. Vice versa a hostile party can overhear and manipulate all communications of a `<<public>>` channel. If neither `<<private>>` nor `<<public>>` are used, we assume by default that the communication channel is not publicly accessible.

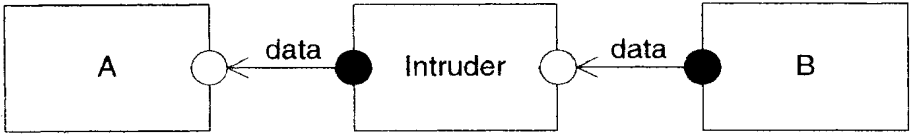
Definition 2 (Private Channel). A `<<private>>` channel has the same semantics as a normal channel within `AUTOFOCUS`, i.e. it is a dedicated connection from one component to another.

Definition 3 (Public Channel). The semantics of a `<<public>>` channel without secrecy and authenticity introduced in Section 3.6, is defined by the SSDs shown in Figure 2. Using a `<<public>>` channel (Figure 2(a)) is an abbreviation for having an intruder included in the model that has access to the channel (Figure 2(b)). The behaviour of the intruder is defined by the threat model—for example, the intruder usually can overhear, intercept or redirect messages. It is possible to model this behaviour in a flexible way using using `AUTOFOCUS` State Transition Diagrams (STDs) (Wimmel and Wisspeintner, 2000).

The identification of private and public channels should be done during the analysis phase of the system development process, right after the identification of the security critical parts. Every security critical channel must be analyzed with regard to accessibility—for the other channels, this is optional. The result of this analysis is documented by using the tags for private and public channels.



(a) <<public>> Channel



(b) Defined as an Intruder between the Two Agents

Figure 2 Semantics of a public Channel

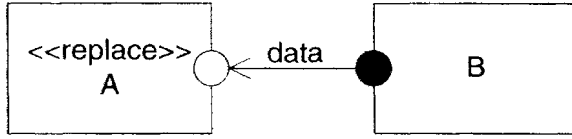
3.3. REPLACEABLE COMPONENTS

Conventional system structure diagrams always show a system in normal operation, e.g. an ordinary electronic purse card component with the specified functionality communicating with the point-of-sale component. In addition to manipulation of the communication link (man-in-the-middle attack), another attack scenario is imaginable: the attacker could try to communicate with the rest of the system *in place of* the purse card. In this case, there is no ordinary purse card in the system, but a faked one (that in particular usually does not contain private keys of ordinary cards, except if they leaked).

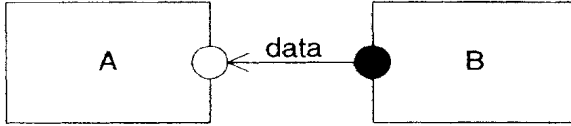
We mark components that can be replaced by faked ones by the attacker with the <<replace>> tag, and components that can not be replaced with the <<nonreplace>> tag. If neither <<replace>> nor <<nonreplace>> is used for a component, the component is non replaceable by default.

Definition 4 (Replaceable Component). Figure 3 shows the semantics of a <<replace>> component. Using a replaceable component (Figure 3(a)) is an abbreviation for specifying two different system scenarios. The first scenario describes the structure of the system with the specified component A (Figure 3(b)). In the second scenario (Figure 3(c)) the attacker exchanges the component by another component A'. A' has the same component signature like A but has an arbitrary behaviour that can be defined by the threat model.

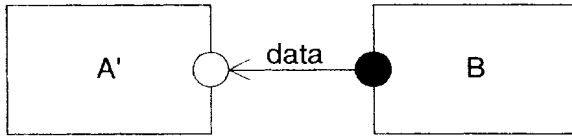
In the development process, replaceable components should be identified during the analysis phase together with the identification of private and public



(a) <<replace>> Component



(b) Scenario 1: Component not Exchanged



(c) Scenario 2: Component Exchanged

Figure 3 Semantics of a Replaceable Component

channels. It is only necessary to analyze security critical components with regard to replaceability.

3.4. ENCAPSULATED COMPONENTS

An encapsulated component is a component that only consists of not publicly accessible subcomponents. In this way an attacker has no possibility to manipulate or exchange the subsystems of this component. Furthermore, the communication within the component cannot be overheard. The security tag <<node>> is used to mark a component as an encapsulated one. The identification of encapsulated components is done together with the identification of private and public channels and replaceable components.

Definition 5 (Consistency Condition of <<node>>). A <<node>> component only consists of <<private>> channels and <<nonreplace>> components.

One example for an encapsulated component is an automated teller machine (ATM). An ATM is encapsulated in a way that unauthorized persons are not

able to manipulate system parts. Overhearing the internal communication is also not possible.

3.5. ACTOR COMPONENTS

Most systems interact with their system environments. It is often desired to illustrate the system environment in the graphical system design. Components that are not part of the system are called actors. We point out actors by using the tag `<<actor>>`. A typical example for an actor is a system user. The system user interacts with the system without being part of it.

Actor components can never be marked with the `<<critical>>` tag. An actor is not part of the system and therefore there is no need to analyze the actor itself with respect to security aspects. But an actor can interact with our system in a way that affects our security requirements. To visualize these critical interactions, channels between actors and the system components can be annotated with the `<<critical>>` tag.

3.6. SECRECY AND AUTHENTICITY

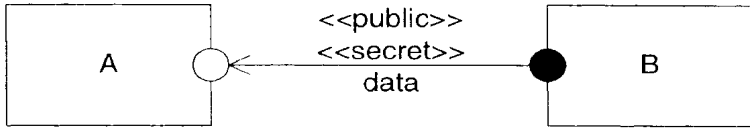
The most important security properties of communication channels—in addition to integrity, availability and non-repudiation—are authenticity and secrecy. For this purpose, we will introduce tags `<<secret>>` and `<<auth>>` for channels in the SSDs. The security properties of channels are identified in the high-level design phase, taking place after the activities of the analysis phase. It is only necessary to specify security properties for security critical, public channels.

There are many possible definitions for authenticity and secrecy in the security literature (see (Gollmann, 1996)). In the following, we give a definition based on our model.

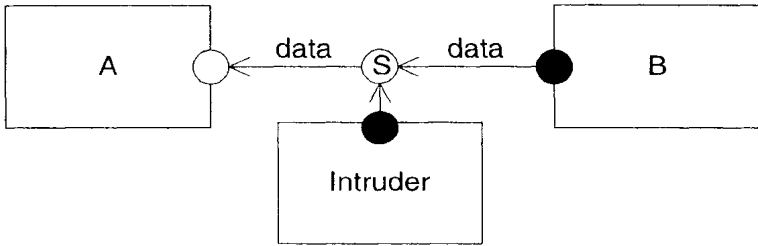
During the high-level design phase, we assume that the defined requirements of secrecy and authenticity are fulfilled automatically, if the corresponding tags appear on the channels. Consequently these requirements restrict the possibilities of an attacker. In the low-level design, the validity of these requirements has to be ensured by proper mechanisms.

Definition 6 (Secret Channel in High-Level Design). A message sent on a `<<secret>>` channel can only be read by its specified destination component. Therefore, we can assume in high-level design that a `<<secret>>` and `<<public>>` channel can not be read by the intruder. But the intruder could write something on it. Figure 4 shows the semantics of a secret and public channel.

Definition 7 (Authentic Channel in High-Level Design). A message received on an `<<auth>>` channel can only come from its specified source component. Therefore, an `<<auth>>` and `<<public>>` channel can not be written



(a) <<secret>> and <<public>> Channel



(b) Intruder can Write Data*

* The "S" circle represents a switch component that distributes all incoming data to all outgoing channels.

Figure 4 Semantics of a Secret and Public Channel

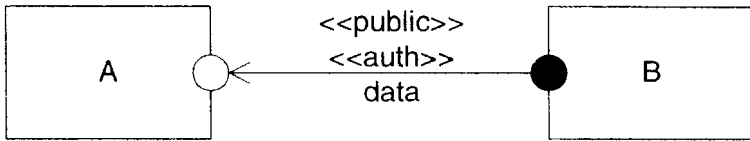
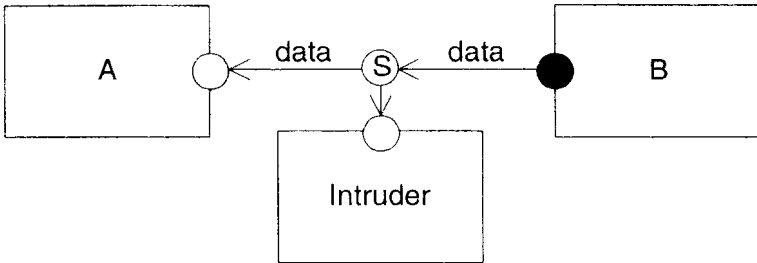
by the intruder. But the intruder could possibly read data from it. Figure 5 illustrates the semantics of an authentic channel.

There are some relations between our notion of security critical and secrecy and authenticity. A security critical channel references to data that should be protected against attackers (see Definition 1). Secrecy and authenticity are security properties. These security properties defines concrete requirements concerning the protection of data against attackers.

Definition 8 (Consistency Condition of <<secret>> and <<auth>>). If a channel is marked to be security critical and the communication is visible for an attacker, the data sent via the channel must be protected in a suitable manner. In this case, during the high-level design phase the protection of data must be ensured by a security property. A <<critical>> and <<public>> channel must be <<secret>> or <<auth>> or both.

3.7. INTEGRITY

We assume that <<secret>> or <<auth>> channels also provide message integrity, i.e. a message received on an <<auth>> channel is guaranteed not to

(a) `<<auth>>` and `<<public>>` Channel

(b) Intruder can Read Data

Figure 5 Semantics of an Authentic and Public Channel

have been modified. In future, the integrity property could also be modelled separately.

3.8. CHANNEL REQUIREMENTS IN LOW-LEVEL DESIGN

In the low-level design phase, taking place after the high-level design phase, the system specification is refined. In this phase, security functionalities can be used to ensure the security properties and requirements.

We can define the usage of symmetric encryption, asymmetric encryption and corresponding signatures to realize the requirements of secrecy and authentication. The security tag `<<sym>>` marks a channel. It defines that the realization of the channel must use a symmetric encryption algorithm to ensure the secrecy requirements. Furthermore the `<<asym>>` tag is used to define the usage of an asymmetric encryption algorithm.

It is also possible to specify the encryption algorithm that should be used to guarantee secrecy. The security tag `<<encalg <Name> [Parameters]>>` can be used together with a channel to specify the usage of a specific encryption algorithm. We can specify additional parameters of the algorithm. For example it is possible to define the key length of the encryption keys. Authenticity can

be realized by using specific authentication protocols. The tag <<authprotocol <Name> [Parameters]>> defines a specific authentication protocol to be used.

By choosing a specific encryption algorithm for realizing secrecy, we perform a refinement step. Special encryption drivers are introduced to perform the encryption and decryption tasks. The data that is sent between the two encryption drivers is encrypted. To realize a specific authentication protocol, special protocol drivers are introduced. Furthermore, additional channels between the protocol drivers are needed to allow bidirectional communication.

After these refinements, the statements on access of the intruder to the channel in Definition 6 and Definition 7 change for <<public>> channels: the intruder now does have read and write access to the channel. The encryption mechanism on the channel must ensure that the intruder cannot manipulate the channel in an improper way, so that the security requirements stated by <<secret>> and <<auth>> are still fulfilled.

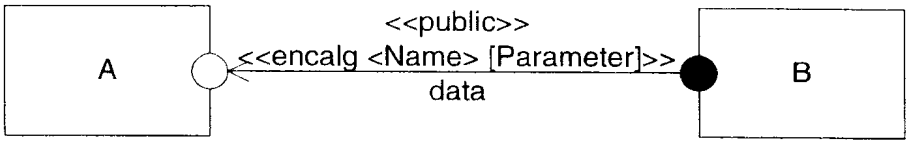
Definition 9 (Secret and Authentic Channels in Low-Level Design). In low-level design, <<public>> channels implementing secret or authentic communication have to be modelled by including appropriate security functionality.

A convenient way to do this is using *security patterns*. Security patterns are generic solutions for common security problems. Figure 6 shows such a pattern for the simple case of encryption (guaranteeing secrecy). The communicating components now include protocol drivers for encryption and decryption of the messages, so the original channel is replaced by an encrypted one. This encryption pattern could be extended by including protocol drivers for key agreement, a public key server or a whole public key infrastructure. Other security patterns, for instance providing auditing functionality, are possible.

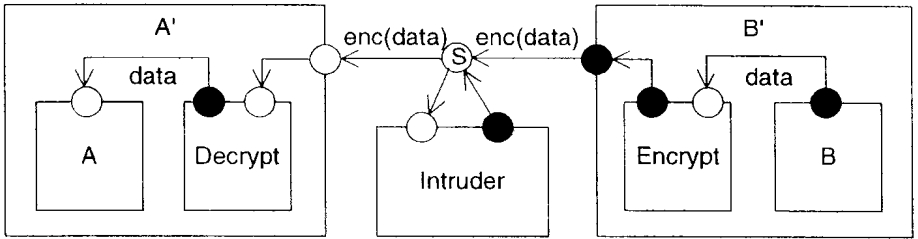
4. EXAMPLE—ELECTRONIC PURSE TRANSACTION SYSTEM

In the previous section, we have introduced extensions for SSDs to deal with security requirements. Now, we use the extended SSDs to model an example system from the area of E-Commerce, an electronic purse card system. The given specification conforms with the *Common Electronic Purse Specification* (CEPS), an international standard for an electronic purse environment (CEP-SCO, 2000). Figure 7 shows the complete system structure of the electronic purse system consisting of several sub-components.

The system user possesses an electronic purse card. The card has some amount of money stored on it and the user can pay with the card at a dealer using a point-of-sale (POS) terminal. The electronic purse card is involved in several security requirements. For example, together with other components the electronic purse card must ensure that an attacker can not steal any money



(a) <<public>> Channel using a Specific Encryption Algorithm



(b) Intruder can Read and Write Encrypted Data

Figure 6 The Encryption Security Pattern

from the whole transaction system. Therefore the card is a security critical component (<<critical>>). Furthermore, the purse card should be realized by a single integrated chip. It is an encapsulated component (<<node>>) and consequently we assume an attacker has no possibility to overhear or manipulate the communication within the electronic purse card. An attacker could try to produce a faked card and he could replace the valid card by the faked one. The security tag <<replace>> of the component ElectronicPurse visualizes this possibility.

The electronic purse card can communicate with the other components over its interface. It offers ports to read the balance, decrease the balance and set the balance to a specific value. The `getBal` operation of the card is used by the POS, a card loading terminal at the bank of the card owner (`IssuerBank`) and by the `CardReader` component. The operation of reading the balance is not security critical because everybody who possesses an electronic purse is allowed to read the stored balance of the card. Consequently the channels `getBal` and `returnBal` do not have the <<critical>> tag. But an intruder could build a special adapter to overhear all communication between the electronic purse card and the other components. To express this possibility the communication channels `getBal`, `returnBal`, `decreaseBal` and `setBal` are marked with the <<public>> tag.

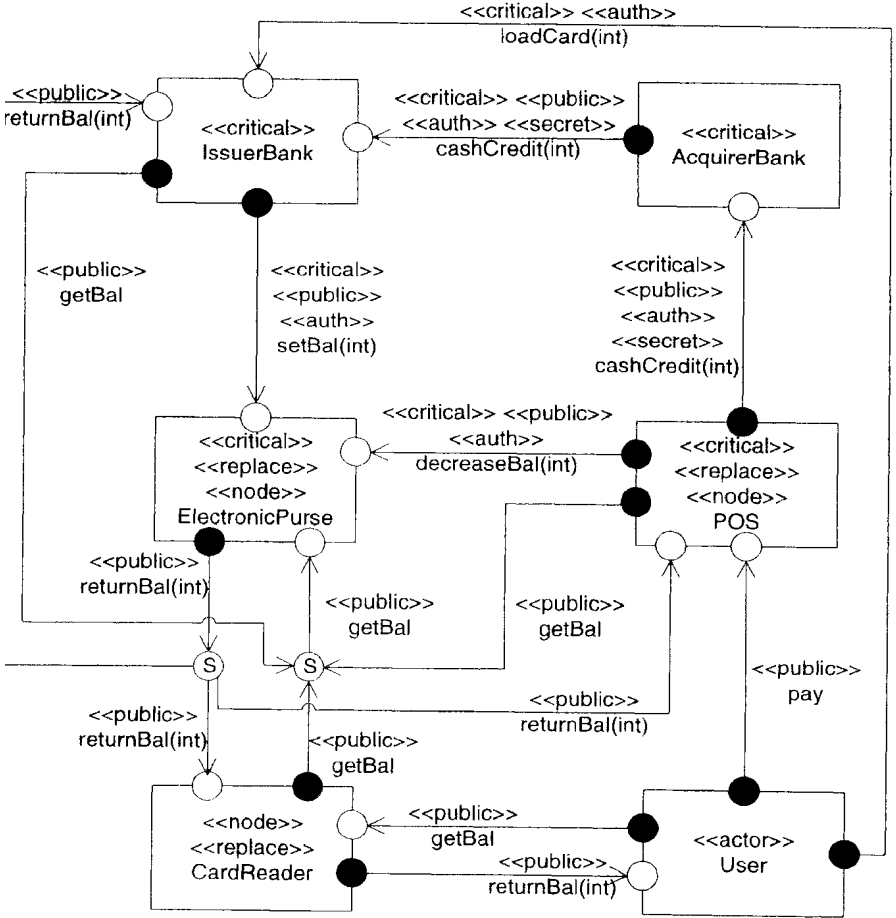


Figure 7 Electronic Purse Card System

If the user wants to load money onto his card, he must go to his bank (IssuerBank) and use a special card terminal. The channel `setBal` is used to transfer money from his bank account to the card. This channel is security critical, because it affects the security requirement about an attacker stealing money. We use the property of authentication (`<<auth>>`) to ensure that the card and the card terminal are valid.

The card reader is a simple device for the user to check the amount of money that is stored on the card. An attacker can exchange the card reader by another component (`<<replace>>`). On the other hand the card reader component is encapsulated (`<<node>>`). No security requirements are defined for this component and therefore the component is not security critical.

The POS component is a card terminal located at a dealer. The user can insert the electronic purse card into the terminal in order to pay for goods. The POS is directly involved in the money transaction process. Therefore this component is security critical. The POS is encapsulated to prevent manipulations within the unit. But a malicious dealer has the possibility to exchange the POS terminal by a faked unit.

The POS component can instruct the electronic purse card to decrease its balance by a given amount. The operation `decreaseBal` is part of the money transaction process and the communication to perform the operations is security critical. To comply with our security requirement that an intruder cannot steal any money, we must ensure that only a valid POS is able to decrease the amount of money on a money card. The POS must authenticate itself in a way that the card is sure to communicate with a valid POS. This fact is visualized in the SSD by the `<<auth>>` tag annotated to the `decreaseBal` channel.

The dealer can submit the earned money to his bank (AcquirerBank) using the `cashCredit` channel. This channel is security critical. The communication medium is a standard telephone line and the potential attacker has possibilities to overhear and manipulate the communications. This fact is expressed using the `<<public>>` tag on this channel. To ensure that an attacker can not overhear or manipulate the transferred data, the `<<secret>>` and `<<auth>>` tags are used.

The channel `cashcredit` between the two banks is used to transfer money from one bank to the other. The communication takes place via an open network (`<<public>>` channel). We must ensure secrecy and authenticity for this channel to protect the transactions data. Both bank components are security critical, but we do not see a risk that a potential attacker can act as a faked bank component. Thus the `<<replace>>` tag is not used for both banks.

Finally let us have a look at the `User`. The user is not part of our system. Therefore the `<<actor>>` tag is used. The user can initiate the paying process at a POS. The initiation of the paying process is not security critical (it just corresponds to inserting the card, whereas the amount of money to be withdrawn is negotiated outside of the system). The critical part of the paying process takes

place between the money card and the POS. Furthermore, the user can load the card with some amount of money. During this action, an amount of money is transferred from his bank account onto the card. This operation is security critical because an attacker could try to transfer money from a foreign account to his own electronic purse card. Thus we need some kind of authentication to perform this operation, e.g. the user must enter a PIN code before the money transaction is performed.

5. CONCLUSIONS AND FURTHER WORK

This work is only the beginning of an effort to extend graphical description techniques for distributed systems with security aspects to support methodical development of security critical systems. We used the CASE tool AUTOFOCUS, the description techniques of which are related to UML-RT, for its simplicity and clear semantics and the possibility to give our security extensions a straightforward and unambiguous meaning.

We showed how to extend AUTOFOCUS system structure diagrams by security tags, both for high-level and low-level design. The transition from high-level to low-level design is aided by the possibility to use security patterns. The description techniques were illustrated with the help of an example from the field of E-Commerce, an electronic purse card system.

We focused on the consideration of channels and system structure. In future, additional security properties such as integrity and availability are to be included. The specification of channels and components in low-level design needs to be detailed, using classifications as pointed out in (Eckert, 1998). Besides, it seems very promising to further examine security patterns providing generic architectures for specific security functionality and evaluate their use within the development process. The refinement of security requirements and security functionalities together with its influence on correctness verification is also part of our research activities.

Also, state transition diagrams (STDs) specifying the behaviour of a component can be extended in a similar way with security properties. For this purpose, it suggests itself to classify the data received and sent on the ports and to use models such as Bell-LaPadula or non-interference—similar as it is done in (Jürjens, 2001) for Statechart diagrams. When the behaviour of components is specified, formal proofs can be carried out (by hand or automatically via model checking) that the specified security properties are fulfilled.

EETs (extended event traces) can also be enriched by cryptographic primitives and security properties, and thus be used to specify and verify security functionality of a component. Examining software development of security critical systems with the help of AUTOFOCUS EETs (using protocols from the CEPS purse card system as a case study) is subject of ongoing work.

Acknowledgments

Helpful comments and encouragement from Oscar Slotosch are gratefully acknowledged.

References

- Bell, D. E. and LaPadula, L. (1973). Secure computer systems: Mathematical foundations and model. Technical Report M74-244, The MITRE Corp., Bedford MA.
- Broy, M., Dederich, F., Dendorfer, C., Fuchs, M., Gritzner, T., and Weber, R. (1992). The Design of Distributed Systems—An Introduction to FOCUS. Technical Report TUM-I9202, Technische Universität München.
- Broy, M. and Slotosch, O. (1999). Enriching the Software Development Process by Formal Methods. In *Current Trends in Applied Formal Methods 1998*, pages 44–61.
- Burrows, M., Abadi, M., and Needham, R. (1989). A logic of authentication. *Proceedings of the Royal Society of London A*, 426:233–271.
- CEPSCO (2000). Common electronic purse specifications: Business requirements. Version 7.0, available from <http://www.cepsco.com>.
- Common Criteria (1999). Common criteria for information technology security evaluation version 2.1. Technical report, Communications Security Establishment (Canada), Service Central de la Sécurité des Systèmes d'Information (France), Bundesamt für Sicherheit in der Informationstechnik (Germany), Netherlands National Communications Security Agency, Communications-Electronics Security Group (United Kingdom), National Institute of Standards and Technology (United States), National Security Agency (United States).
- Eckert, C. (1998). *Sichere, verteilte Systeme – Konzepte, Modelle und Systemarchitekturen*. professorial thesis, Technische Universität München.
- Goguen, J. A. and Meseguer, J. (1998). Security Policy and Security Models. In *Proceedings of 1982 IEEE Symposium on Security and Privacy*.
- Gollmann, D. (1996). What do We Mean by Entity Authentication? In *Proceedings of 1996 IEEE Symposium on Security and Privacy*.
- Huber, F., Molterer, S., Rausch, A., Schätz, B., Sihling, M., and Slotosch, O. (1998a). Tool supported Specification and Simulation of Distributed Systems. In *International Symposium on Software Engineering for Parallel and Distributed Systems*, pages 155–164.
- Huber, F., Molterer, S., Schätz, B., Slotosch, O., and Vilbig, A. (1998b). Traffic Lights—An AutoFocus Case Study. In *1998 International Conference on Application of Concurrency to System Design*, pages 282–294. IEEE Computer Society.

- ITSEC (1990). ITSEC. Information Technology Security Evaluation Criteria—Harmonised Criteria of France, Germany, the Netherlands, the United Kingdom. Version 1.
- ITU (1996). ITU-TS Recommendation Z. 120: Message Sequence Chart (MSC). ITU-TS, Geneva.
- Jones, M. P. (August 1993). *An Introduction to Gofer*.
- Jürjens, J. (2001). Towards Development of Secure Systems using UML. In *FASE '01: Fundamental Approaches to Software Engineering*. to appear.
- Lotz, V. (2000). Formally Defining Security Properties with Relations on Streams. In Schneider, S. and Ryan, P., editors, *Electronic Notes in Theoretical Computer Science*, volume 32. Elsevier Science Publishers.
- Lowe, G. (1996). Breaking and fixing the Needham-Schroeder Public-Key Protocol using FDR. In Margaria and Steffen, editors, *TACAS*, volume 1055 of *lncs*, pages 147–166. sv.
- Paulson, L. C. (1998). The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1–2):85–128.
- Philipps, J. and Slotosch, O. (1999). The Quest for Correct Systems: Model Checking of Diagramms and Datatypes. In *Asia Pacific Software Engineering Conference 1999*.
- Slotosch, O. (1998). Quest: Overview over the Project. In Hutter, D., Stephan, W., Traverso, P., and Ullmann, M., editors, *Applied Formal Methods—FM-Trends 98*, pages 346–350. Springer LNCS 1641.
- Thayer, F., Herzog, J. C., and Guttman, J. D. (1998). Strand Spaces: Why is a security protocol correct? In *Proceedings of 1998 IEEE Symposium on Security and Privacy*.
- Thompson, S. (1999). *Haskell: The Craft of Functional Programming*. Addison-Wesley Longman.
- Wimmel, G., Lötzbeyer, H., Pretschner, A., and Slotosch, O. (2000). Specification Based Test Sequence Generation with Propositional Logic. *Journal on Software Testing Verification and Reliability*, 10:229–248.
- Wimmel, G. and Wisspintner, A. (2000). The Needham-Schroeder Protocol—an AutoFocus Case Study. Internal report, Technische Universität München.