Audun Jøsang
Torleiv Maseng
Svein Johan Knapskog (Eds.)

LNCS 5838

Identity and Privacy in the Internet Age

14th Nordic Conference on Secure IT Systems, NordSec 2009
Oslo, Norway, October 2009
Proceedings

Springer

Lecture Notes in Computer Science 5838

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

Audun Jøsang Torleiv Maseng
Svein Johan Knapskog (Eds.)

Identity and Privacy in the Internet Age

14th Nordic Conference on Secure IT Systems, NordSec 2009
Oslo, Norway, 14-16 October 2009
Proceedings

Springer

Volume Editors

Audun Jøsang
University of Oslo
University Graduate Center
Kjeller, Norway
E-mail: josang@unik.no

Torleiv Maseng
Norwegian Defence Research Establishment
Kjeller, Norway
E-mail: Torleiv.Maseng@ffi.no

Svein Johan Knapskog
Norwegian University of Science and Technology
Centre for Quantifiable Quality of Service
in Communication Systems
Trondheim, Norway
E-mail: Knapskog@q2s.ntnu.no

Library of Congress Control Number: 2009935066

CR Subject Classification (1998): D.4.6, K.6.5, D.2, H.2.7, K.4.4, E.3, C.2

LNCS Sublibrary: SL 4 – Security and Cryptology

ISSN 0302-9743
ISBN-10 3-642-04765-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-04765-7 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12769834 06/3180 5 4 3 2 1 0

Preface

The NordSec workshops were started in 1996 with the aim of bringing together researchers and practitioners within computer security in the Nordic countries – thereby establishing a forum for discussions and co-operation between universities, industry and computer societies. Since then, the workshop has developed into a fully fledged international information security conference, held in the Nordic countries on a round robin basis. The 14th Nordic Conference on Secure IT Systems was held in Oslo on 14-16 October 2009. Under the theme Identity and Privacy in the Internet Age, this year's conference explored policies, strategies and technologies for protecting identities and the growing flow of personal information passing through the Internet and mobile networks under an increasingly serious threat picture. Among the contemporary security issues discussed were security services modeling, Petri nets, attack graphs, electronic voting schemes, anonymous payment schemes, mobile ID-protocols, SIM cards, network embedded systems, trust, wireless sensor networks, privacy, privacy disclosure regulations, financial cryptography, PIN verification, temporal access control, random number generators, and some more.

As a pre-cursor to the conference proper, the Nordic Security Day on Wednesday 14 October hosted talks by leading representatives from industry, academia and the government sector, and a press conference was given.

On Thursday 15, a keynote talk was given by Simone Fischer-Hübner, University of Karlstad, Sweden, on the currently strongly researched topic of identity management, entitled *Privacy-Enhancing Identity Management – Challenges for the Future* and on Friday 16 October, a keynote entitled: *Identity Management on the Internet: Opportunities and Challenges for Mobile Operators* was given by Do van Than, Senior Research Scientist at Telenor R&I and Professor at the Norwegian University of Science and Technology (NTNU).

The conference received 52 valid submissions, and 20 regular papers were chosen for presentation and inclusion in the LNCS proceedings by Springer. In addition 8 short papers were chosen for presentation at the conference. All papers were peer reviewed by at least two reviewers, most papers by three, and some by four.

We would like to thank all the people who helped in making the 14th Nordic Conference on Secure IT Systems a scientific and social success. First, we commend the Steering Committee for the decision to allow the conference to be held at the University of Oslo, and for their advice on the general format of the event. Secondly, a heartfelt thanks goes to the Program Committee members and the sub-reviewers who spent valuable time reviewing and discussing the review results. The smooth submission and review processes were run by the Springer OCS, giving invaluable support to the PC, in particular the PC Chair. We also thank the staff at Springer for their advice and support during the preparation of the proceedings.

And last, but not least, we thank the Organizing Committee for the impeccable organization and running of the conference.

October 2009

Audun Jøsang
Torleiv Maseng
Svein Johan Knapskog

Organization

General Chair

Audun Jøsang	University of Oslo – University Graduate Center, Kjeller, Norway

Program Co-chairs

Svein Johan Knapskog	Norwegian University of Science and Technology Center for Quantifiable Quality of Service in Communication Systems, Trondheim, Norway
Torleiv Maseng	Norwegian Defence Research Establishment, Kjeller, Norway University of Lund, Lund, Sweden

Program Committee

Viiveke Fåk	Linköping University, Sweden
Dieter Gollman	University of Technology Hamburg-Harburg, Germany
Peeter Laud	University of Tartu, Estonia
Nahid Shahmehri	Linköping University, Sweden
Simone Fischer-Hübner	Karlstad University, Sweden
Tuomas Aura	Microsoft Research, Cambridge, UK
André Zúquete	University of Aveiro, Portugal
André Årnes	Oracle Norway
Björn Pehrson	KTH – Royal Institute of Technology, Sweden
Chris Mitchell	University of London – Royal Holloway, UK
Christian Damsgaard Jensen	Technical University of Denmark, Denmark
Christian Larsen	National Criminal Investigation Service, Norway
Chunming Rong	University of Stavanger, Norway
Danilo Gligoroski	Norwegian University of Science and Technology, Norway
Denis Trcek	University of Ljubljana, Slovenia
Dogan Kesdogan	University of Siegen, Germany
Ed Dawson	Queensland University of Technology, Australia
Eldfrid Øvstedal	SINTEF, Trondheim, Norway
Eli Winjum	Norwegian Defence Research Establishment, Norway
Erik Hjelmås	Gjøvik University College, Norway

Geir Hallingstad	NATO
George Polyzos	Athens University of Economics and Business, Greece
Inger Anne Tøndel	SINTEF, Trondheim, Norway
Jennifer Seberry	University of Wollongong, Australia
Joakim von Brandis	Mnemonic, Norway
Kåre Presttun	Mnemonic, Norway
Josef Pieprzyk	Macquarie University, Australia
Kai Rannenberg	T-Mobile, Germany
Karin Sallhammar	Telenor, Norway
Knut Johannessen	Telenor, Norway
Lawrie Brown	Australian Defence Force Academy, Australia
Patrick Bours	Gjøvik University College, Norway
Peter Herrmann	Norwegian University of Science and Technology, Norway
Richard Kemmerer	University of California, Santa Barbara, USA
Simin Nadjm-Tehrani	Linköping University, Sweden
Stefan Lindskog	Karlstad University, Sweden
Tomas Olovsson	Chalmers University, Sweden
Tor Helleseth	University of Bergen, Norway
Vlastimil Klima	Independent cryptologist, Czech Republic
Vaclav Matyas	Masaryk University, Czech Republic
Vladimir Oleshchuk	University of Agder, Norway
Wei Wang	INRIA, France
Willy Susilo	University of Wollongong, Australia
Yuliang Zheng	University of North Carolina at Charlotte, USA
Zhili Sun	University of Surrey, UK
Svein Yngvar Willassen	Norwegian University of Science and Technology, Norway
Vijay Varadharajan	Macquarie University, Australia
Mads Dam	Royal Institute of Technology, Sweden
Andrew Clark	Queensland University of Technology, Australia
Anne Karen Seip	The Financial Supervisory Authority of Norway, Norway
Kristian Gjøsteen	Norwegian University of Science and Technology, Norway
Sven Laur	Helsinki University of Technology, Finland
Åsmund Skomedal	Norwegian Computing Center, Norway
Maria Line	SINTEF, Norway

Sponsors

Telenor Group
Nisnet
University of Oslo – University Graduate Center
Norwegian Computing Center

Table of Contents

Session 1: Anonymity and Privacy

Session 2: Modelling and Design

Session 3: Network Layer Security

Session 4: Security for Mobile Users

Session 5: Embedded Systems and Mechanisms

Session 6: Protocols and Protocol Analysis

On the Effectiveness of Privacy Breach Disclosure Legislation in Europe: Empirical Evidence from the US Stock Market

Jan Muntermann[1] and Heiko Roßnagel[2]

[1] Goethe-University Frankfurt, House of Finance,
Grüneburgplatz 1
60323 Frankfurt, Germany
muntermann@wiwi.uni-frankfurt.de
[2] Fraunhofer Institute for Industrial Engineering (IAO),
Nobelstr. 12,
70569 Stuttgart, Germany
Heiko.Rossnagel@iao.fraunhofer.de

Abstract. Several U.S. states have enacted laws that require organizations to notify the affected individuals if personal data under their control is believed to have been acquired by an unauthorized person. In the EU, where a similar legislation is still missing, several researchers have recommended the introduction of a security-breach notification law. The intention of these laws is twofold. On one hand, they should enable affected individuals to take appropriate steps to protect themselves against malicious impacts resulting from the breach. On the other hand, it was intended to create incentives for companies to undertake steps to improve their security measures. In this contribution, we explore these incentives and present an event study in order to examine the effects of privacy incident announcements on the stock prices of affected companies. Our results show that there are significant price reactions on the next day following the announcements. By comparing these price reactions with those observed for other event types, we detect that disclosed privacy incidents are perceived as marginal by the market. The results show that existing disclosure regulation provides little to no incentives to invest in security measures to prevent the occurrence of privacy breaches, since they are widely ignored by the capital markets. From the widely discussed incentive perspective, the privacy breach disclosure legislation does not appear to be effectively addressing this goal.

Keywords: Privacy Breaches, Event Study, Privacy Disclosure Regulation.

1 Introduction

With a growing amount of personal data stored and processed by organizations, the complexity of information systems safeguarding the information also increases [29]. Over the past few years, privacy breaches have been reported frequently. Over 220 million private records have been lost or stolen in the United States since January 2005 [23]. These incidents, that involve the misuse of personal information, may be

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 1–14, 2009.

related to security breaches and may include the illegal sale of customer data or the loss of equipment containing sensitive consumers' or employees' information [1]. One recent example, that received considerable publicity, was the T-Mobile data protection scandal in Germany [42], where personal information of some 17 million mobile phone customers, including politicians, business leaders and TV stars, was stolen. Names, phone numbers, dates of birth and, in some cases, email addresses were stolen, affecting half of the customers of T-Mobile. The theft happened in 2006 but was revealed in October of 2008 by a German news magazine [42].

In order to protect their confidential data as well as the personal information of their customers, organizations need to invest in information security [29]. The high frequency of privacy incidents that have been announced over the last several years raises the question whether organizations have the necessary incentives to safeguard their customers' private information [1] [3]. As can be observed in several cases (e.g. [49]; [42]), even firms that are aware of a privacy breach do not necessarily promptly report the incidents in order to avoid a loss of confidence by customers.

To create these incentives, several U.S. states have recently enacted laws that require organizations to notify the affected individuals if personal data under their control is believed to have been acquired by an unauthorized person [3]. The first state that enacted a privacy breach disclosure law has been California in 2002 [6]. The intention of the law was twofold. On one hand, it should enable affected individuals to take appropriate steps to protect themselves against malicious impacts resulting from the breach. On the other hand, it was intended to create an incentive for companies to undertake steps to improve their security measures [3].

Similar legislation has since then been passed by at least 42 other states [34].This has led to a strong increase in the number of reported incidents. Several research groups are collecting these incidents in public accessible repositories such as the "Chronology of data breaches" [38] or the "Data Loss Archive and Database" [4]. On the basis of the latter database, Fig. 1 shows monthly breaches that were reported in the US from 2000 to the first quarter of 2008.

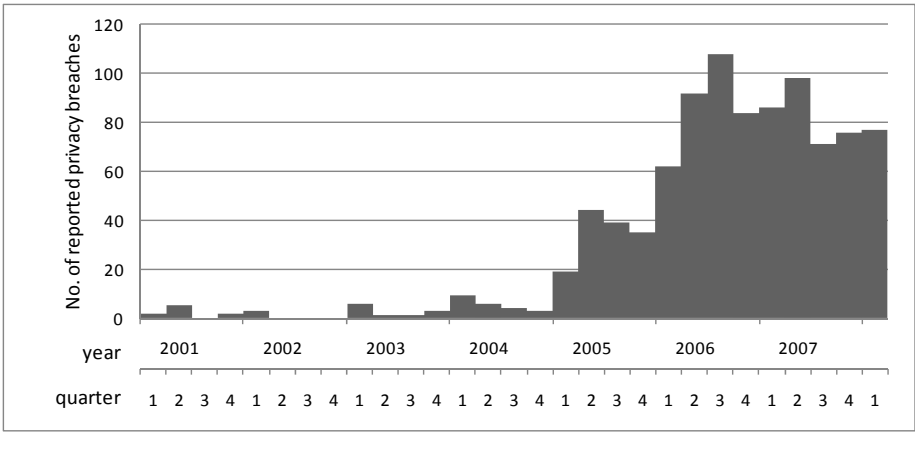


Fig. 1. Reported privacy breaches in the US, 2001-2008Q1 (N=952)

The dramatic increase in reported incidents from 2004 onwards demonstrates the impact of the privacy breach disclosure laws in the US. In the EU, a similar legislation is still missing. Therefore, Anderson et al. [3] and Roussopoulos et al. [39] recommend in studies conducted on behalf of the European Network and Information Security Agency (ENISA) that the EU should introduce a comprehensive security-breach notification law.

The effectiveness of these laws with regard to the improvement of customers' data security can not be demonstrated easily. Privacy breach notification relies on creating incentives for organizations to invest in security measures [3]. These incentives result from the costs organizations endure due to the notifications of the incidents. Immediate punishment can result from a direct penalty by a governing institution. In the US, the Federal Trade Commission (FTC) can fine a firm responsible for a privacy breach as for example the $15 Million fine of Choicepoint in 2006 [18]. Further costs could arise from liability lawsuits being filed by affected individuals. Furthermore, notification of affected customers and providing assistance also lead to direct costs for the organization [37]. In addition to the direct costs, the privacy incident can lead to further long term indirect costs [19]. These costs comprise of loss of existing customers, increased difficulty in recruiting new customers and measures to cover the brand damage [37].

Quantifying these costs is a remarkably complex task [19] [40]. One approach to simplify this task is to measure the impact on the company value to approximate the effects and total costs of privacy incidents. For companies whose shares are traded at an exchange and that are actually being faced with a privacy breach unveiled, stock prices will adjust if the breach is relevant to the company´s reputation and customers´ trust and therefore, to the shareholders value.

In this contribution we present an event study in order to examine if and how privacy incidents have a significant effect on stock prices of companies facing a privacy breach. By comparing these price reactions to other events that lead to abnormal returns such as announcements on changes in the management board [12], IT investments [26], E-Commerce initiatives [43] and dividend and earnings announcements [2] [35], we can gain an insight into how severe these incidents are perceived by the market. This in return can indicate the strength of the incentive to invest in security measures to prevent the occurrence of privacy breaches and therefore provide an indication of the effectiveness of privacy breach disclosure legislation.

This paper is structured as follows. We first present a literature review on related work (section 2). In section 3, we present the theoretical foundation of our research and the research hypotheses we have derived. In section 4, we describe the methodology of our approach and provide a description of the data set. Empirical results are presented in section 5, followed by a discussion of their implications in section 6. The paper concludes with a discussion of the results in section 7.

2 Related Work

Several recent event studies have analysed the impact of *security breach* announcements on the market value of firms [8] [27] [11] [20] [28] [15]. Similar studies have been conducted on denial of service attacks [24], virus attacks [25] and vulnerability

announcements of software vendors [44]. Garg et al. [20] and Cavusoglu et al. [11] found evidence that a firm's announcement of a security breach is negatively associated with its market value. The results of Kannan et al. [28], however, only show a negative but not significant effect. The authors attribute this difference of their results to the difference in the reference group against which the abnormality was computed. While Cavusoglu et al. [11] used the general market index, Kannan et al. [28] chose a specific control firm for each firm in their data set. Campbell et al. [8] found a highly significant negative market reaction following information security breaches involving unauthorized access to confidential data, but no significant reaction when the breach did not involve confidential information, demonstrating that the type of the breach can affect the results. Cavusoglu et al. [11] found evidence that the size and the type of the company also affect the magnitude of the negative abnormal return.

In the study most related to our work, Acquisti et al. [1] analysed the effects of privacy breach announcements on the market value of firms. Their results show a significant negative impact of privacy breaches on the company's market value on the announcement day. They also observe that the cumulative effect increases in magnitude during the first two days, before it decreases and loses statistical significance. However, they do not analyse the magnitude of these price reactions and compare this effect magnitude with those effect magnitudes that can be observed for other event types.

3 Theoretical Background and Hypotheses

3.1 Theoretical Foundation

In the ideal case, prices that can be observed on an efficient market fully reflect all information available to the market participants all the time. This well-known efficient market hypothesis (EMH) and the theory of capital markets go back to [17] and [16] who describe three forms of the efficient market hypothesis. In the context of this work, we focus on the semi-strong form providing the theoretical foundation for the research approach of this paper. Here, according to the theory, asset prices adjust efficiently to new information available and therefore, if there is new and price-relevant information available, a market reaction will be observable subsequent to the event date (if there are not insider or anticipation effects).

For companies whose shares are traded at an exchange and that are actually being faced with a privacy breach unveiled, stock prices will adjust if the breach is relevant to the company´s reputation and customers´ trust, and therefore, to the shareholders value. The semi-strong efficient market hypothesis states that in this situation, stock prices will promptly adjust. This process can be observed and analyzed with an appropriate methodology, so-called event studies that will be introduced in section 4.

Event study analysis represents a methodical framework for testing the efficient market hypothesis, especially for testing the semi-strong form. This is done by analyzing observed stock price adjustments that follow market events such as company announcements or stock splits [16]. The efficient market hypothesis and event study analysis represent a foundation that has been widely used in different research domains. Having their roots in economic and finance research [30], event study analyses

have been conducted since the 1930's to assess the impact of diverse event types that could affect investors' assessments, and therefore, asset prices. In other disciplines, including information systems research, EMH and event study analyses provided the basis for IT-related research, e.g. in order to assess the effect of announcements of newly created CIO positions [12].

3.2 Derived Research Hypotheses

As discussed above, a privacy breach can lead to direct and indirect costs. Privacy breaches may signal to the market that the company is not concerned about customer privacy or that its internal security practices are poor. This may lead investors to question the company's long-term performance [11]. As described in the previous section the expectations of investors are reflected in the market prices. Therefore, if investors view a privacy breach negatively, believing that the costs associated with the breach will substantially reduce expected future cash flows, a negative abnormal return can be expected [1]. This leads to our first research hypothesis:

H1: *A company will experience negative, abnormal changes in market value whenever a privacy breach is announced.*

Cavusoglu et al. [11] found evidence to support their hypotheses, that the size and type of a company affects the magnitude of abnormal changes in market value following a security breach announcement. In order to examine the effects of the company type, they grouped companies into two categories: conventional firms and net firms. Their results show that the magnitude of abnormal negative returns for internet security breaches is larger for net firms than for conventional firms. Based on these results one could expect a similar outcome for privacy incidents. However, we feel that the simple categorization between net firms and conventional firms is too simplistic for the complex implications following privacy incidents. Instead, based on the business subtype category of Attrition.org [4], we grouped companies into 6 different branches: Financial, Retail, Technology, Industry, Data, and Others. Based on this we formulate a second research hypothesis:

H2: *The magnitude of abnormal changes in market value following privacy incidents depend on the business type of the affected firm.*

4 Research Methodology

4.1 Event Study Setup

Event study analysis represents a methodical framework for testing the research hypotheses we have presented in the previous section. In IT-related research, the methodology has been applied successfully in the past, e.g. to analyze the effect of announcements on new CIO positions [12] on IT investments [13] or E-Commerce announcements [43].

When conducting an event study, it is necessary to define an event type of interest and to collect event and stock price data (see the following section 4.2) in order to analyze price effects that might be observable subsequent to the event dates. Then, it

is required to calculate so-called abnormal stock returns, i.e. to adjust the returns observed by a general market trend and/or stock behaviour that could be expected if no event would have been observed. The approach to calculate these abnormal returns needs to be defined by a return-generating model, which formalizes abnormal return calculations. We apply the most commonly applied market-model approach (e.g. see [7] [5]), i.e. abnormal returns $AR_{i,t}$ and cumulative abnormal returns $CAR_{i,t1,t2}$ are defined as:

$$AR_{i,t} = R_{i,t} - (\alpha_i + \beta_i R_{m,t}) \qquad CAR_{i,t1,t2} = \sum_{t=t1}^{t2} AR_{i,t} \tag{1}$$

where :

$AR_{i,t}$	=	Abnormal return of the common stock of company *i* on day *t*
$CAR_{i,t1,t2}$	=	Cumulative abnormal return of the common stock of company *i* from day *t*1 to *t*2
$R_{i,t}$	=	Rate of return of the common stock of company *i* on day *t*
$R_{m,t}$	=	Rate of return of the common stock of market index on day *t*
α_t, β_t	=	OLS estimates of the linear model that describes the sensitivity of $R_{i,t}$ to the market index $R_{m,t}$ (calculated for an estimation windows of 255 days in length that ends 31 days prior to the event date).

Calculated abnormal returns are grouped as return populations that provide the empirical basis for statistical test procedures addressing the research hypotheses presented in the previous section.

4.2 Dataset Description

We use a collection of press releases on privacy breaches that were collected and stored in the Data Loss Archive and Database by Attrition.org, a community-driven project that aims at collecting, disseminating and distributing information about data breaches involving personally identifying information (privacy incidents) [4].

From this database, we extracted those privacy incidents that were documented for companies whose shares are listed and traded publicly at a stock exchange. The sample covers privacy incidents that were observed between 2001-01-01 and 2007-12-31. For those companies, more precisely for the corresponding stocks, daily stock and market index, prices were collected from the respective exchange. We had to discard several announcements due to several reasons: First, our focus is on affected companies whose shares are listed at a stock exchange reducing the number of analyzed incidents from 952 to 142. For the remaining events, it was necessary to collect price series for an estimation window of 255 days plus an analysis window of 11 days. As a result, our sample comprises 97 privacy breach announcements for which the corresponding stock price series were available for further analysis.

5 Empirical Results

Applying the return-generating model to the return series collected, we calculated abnormal returns for an analysis window of 5 trading days prior to and 5 trading days

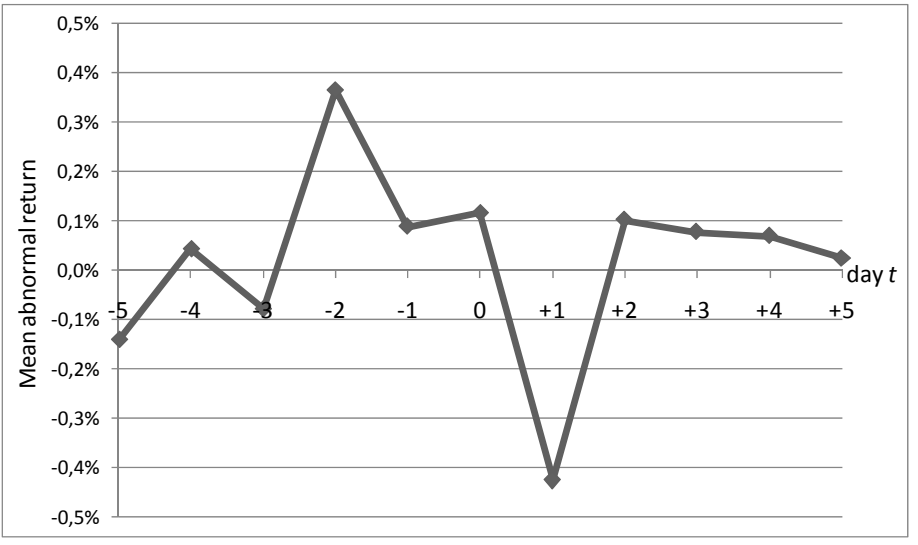


Fig. 2. Mean abnormal returns from 5days prior to and 5 days subsequent to the event at t=0

subsequent to the event dates. As a result, we obtain abnormal return samples for each day within the analysis window where day $t = 0$ is defined as event date. Sample means, i.e. the average abnormal price effects of these abnormal return samples are illustrated in Fig. 2 showing a negative peak on the next day following the event date.

After calculating these samples for each day of the analysis window $t = \{-5, ..., +5\}$, we formulate statistical test hypotheses to address our formulated research hypothesis H1: "A company suffers a loss in market value whenever a privacy breach is announced". Having a sample size of 97 calculated abnormal returns per analysis day, we applied a parametric test procedure. For each day t of the analysis window $t = \{-5, ..., +5\}$ we test whether or not $AR_{i,t}$ is significant less than zero (H0: Mean($AR_{i,t}$) ≥ 0 to be rejected). Test results are shown in the following table 1.

Table 1. Price reactions to privacy breach announcements

	Abnormal returns										
Day t	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5
Mean AR_t in %	-0.14	0.04	-0.08	0.37	0.09	0.12	-0.42*	0.10	0.08	0.07	0.02
t-value	-0.93	0.39	-0.38	1.22	0.41	0.38	-1.67*	0.66	0.31	0.43	0.13
	Cumulative abnormal returns										
Period ($t1$; $t2$)	(0; 1)		(1; 2)		(1; 3)		(1; 4)		(1; 5)		
Mean $CAR_{i,t1,t2}$ in %	-0.31		-0.32		-0.25		-0.18		-0.15		
t-value	-0.93		-1.29		-0.76		-0.54		-0.37		

* indicates significance at the 5% level (one-tailed test).

We find no evidence for abnormal negative price reactions in the period of five days prior to the event date. Second, we examine the ex-post price effect. For the days following the event date, we detected a significant negative abnormal return for t = +1, i.e. the day following the event date, an observation which provides strong support for H1. This is remarkable, since there seems to be a delay at which stock prices react to the announcement of privacy breaches. This finding is in contrast to previous research on price effects that can be observed for company announcements having a regulatory background, i.e. where companies satisfy existing law. Here, prompt price reactions have been observed in the past (e.g. [35]; [9]; [32]) The delay of price reaction we observed is also in contrast to Acquisti et al. [1] who observed significant abnormal returns for the event dates. We assume that this observation results from the different event collection procedures. Their dataset (i.e. the dates of the privacy incident announcements) was collected from general news databases like Lexis-Nexis and ProQuest that contain print media. These incident announcements should have been published online the day before. Some empirical examples corroborate this statement: A data breach at VeriSign which is listed at the 6th of August 2007 in our dataset (which stems from an online source) can be found in the Lexis-Nexis database at the 7th of August (from The Associated Press newswire) and at the 8th of August where American Banker reported on a laptop with Social Security numbers that was stolen from an employee of Verisign [48]. In a second case, our dataset contains an unauthorized breach of privacy data from Pfizer that was published online (at pharmalot.com) on the 11th of June 2007. The same event can be found on Lexis-Nexis for the 12th of June 2007 from the original source The Star-Ledger newspaper [33].

Besides the temporal dimension, the price effect's magnitude is of major relevance. In order to evaluate the effectiveness with regard to creating incentives to invest in security measures, showing the significance of the observed price effects is not enough. In addition, we have to observe and asses the magnitude of the price effects. Therefore, we compared the average of significant abnormal returns we have observed with other event studies on company announcements that were published due to regulation and which (for comparability reasons) also applied the market model as return-generating model. The -0.42 % that we observed on average, represent a comparatively small effect. Aharony and Swary [2] detect a significant average abnormal return of -2.30 % for dividend decreases that were reported in quarterly earnings and dividend announcements. Sanger and Peterson [41] observed a significant average abnormal return of -8.51 % for stocks for which a delisting from a stock exchange has been announced. For announcements on E-Commerce investments, Subramani and Walden [43] find cumulative abnormal returns of 7.5 %.

Compared to these empirical results, the -0.42 % average abnormal price reaction we have detected for news on privacy incident seems comparatively small. The magnitude of this price effect is even lower compared to price effects observed for announcements of newly created CIO positions for which Chatterjee et al. [12] observed a significant abnormal (positive) return of 0.49 %.

For the cumulative abnormal returns we have calculated for different periods following the event date, we have not detected any significant cumulative price effect. This is in contrast to Acquisti et al. [1] who observed a significant negative cumulative price effect for the (0; +1) period. However, most of the price effect originates

from the effect that has been observed on the event date. We can conclude that the empirical results do not provide evidence that there is a long term price effect.

In order to address research hypothesis H2: *"The magnitude of abnormal negative returns for privacy depends on the business type of the affected firm"*, we generated sub-samples for different business types that comprise: Financial, Retail, Technology, Industry, Data, and Others. Each sub-sample contains those abnormal returns $AR_{i,+1}$ for which we observed an abnormal price reaction, i.e. the day subsequent to the event date (t = +1). For each business type, we compare theses business type sub-samples with the remaining total sample. Since we have sub-sample sizes smaller 30, we apply the Wilcoxon Rank Sum test for comparing two independent samples (H0: Median($AR_{business\ sub\text{-}sample}$) ≠ Median($AR_{remaining\ total\ sample}$) to be rejected) - a non-parametric statistical test procedure for comparing the median of smaller data samples [47]. Test results are presented in the following table 2.

Table 2. Price reactions for different business types

	Business type sub-samples					
	Financial	Retail	Technology	Industry	Data	Others
Median AR_{+1} in %	-0.08	-0.25	-0.39	0.43	-0.92	-0.40
Sub-sample size *n*	28	18	20	14	4	13
	Remaining total sample					
Median AR_{+1} in %	-0,33	-0.28	-0.20	-0.30	-0.24	-0.27
N - n	69	79	77	83	93	84
Wilcoxon z-value approx.	0,11	0,12	-0,42	1,86*	-1,27	-0,95

* indicates significance at the 10 % level (two-tailed test).

We detect a significant difference for the Industry segment, which shows significantly less negative price behaviour than the other segments at a .1 level of significance. Furthermore, it is remarkable that in contrast to all other business types, the Industry segment does not face a negative abnormal market reaction in more than 50% of the cases observed in our sample.

For the Data business type we find some evidence that this segment seems more affected than the other segments. Unfortunately, the sub-sample size is quite small limiting its significance. However, the *Data* sub-sample shows the lowest AR_{+1} Median of -0.92 %.

The empirical results provide some evidence that depending on a company's business type, the changes in market value due to published privacy incidents is stronger or less strong compared to other business types, providing weak support for H2.

6 Implications

6.1 Regulatory Implications

The dramatic increase in reported incidents from 2004 onwards demonstrates the impact of the security breach disclosure laws in the US, which has lead researchers to

recommend a similar legislation for the European Union [3] [39]. The number of reported incidents provides support that a higher transparency of privacy incidents will be achieved. This transparency is the prerequisite for enabling affected individuals to take appropriate steps to protect themselves against malicious impacts resulting from a breach, which is one of the goals of privacy disclosure legislation. Furthermore, we detected a significant negative abnormal return for the day following the initial press release, which provides strong support that investors react to the announcement of privacy breaches. This finding, however, is in contrast to previous research on price effects for company announcements satisfying existing law, where prompt price reactions have been observed. This suggests that the existing privacy disclosure legislation in the US is able to achieve transparency for affected individuals but not able to provide a full market transparency of privacy incidents. This lack of market transparency is further highlighted by the differences in the dates of the announcements between our sample and the one used in Acquisti et al. [1]. To overcome this Anderson et al. [3] recommend that a central clearing house should be informed as well. They claim that such regulatory requirements would ensure that even minor breaches could be located by the press, by investors (i.e. by general market participants), by researchers, and by sector-specific regulators.

Anderson et al. [3] also claim that privacy disclosure laws create incentives for companies to invest in better security measures using the significant negative price effects observed in Acquisti et al. [1] as an argument. However our results show that the magnitude of these reactions is rather small compared to other incidents such as dividend announcements and announcements of E-Commerce investments. Furthermore, some sectors like the industry sector face an even lower market reaction. This weak market reaction raises major doubts about the effectiveness of the privacy breach disclosure legislation with regard to creating incentives for companies to invest in better security.

Therefore, regulators should consider additional measures to increase the incentives for a better protection of customer's data and customers should consider protecting themselves by using privacy enhancing technologies when possible.

6.2 Managerial Implications

When companies are faced with a security investment decision, the management has to assess the amount of the security investment, which is a challenging task. One possible approach is determined by the costs of the security investment. Using this approach, the decision makers within an organisation define a fixed budget to be spent on security. Than, the responsible management aims at achieving the optimal amount of security for that budget appropriated. Thereby, the benefits of the security investment are simply assumed to be overhead costs and will not be quantified. A limitation of this approach is that it does not help to decide on the optimal amount of money to be spent on security [10].

The alternative is to rely on the traditional risk or decision analysis framework. Therefore, an expected loss is estimated for a security breach on the basis of potential risk factors identified, expected losses and their likelihoods [10] [31] [45] [22]. The major challenge of the approach is to estimate the probabilities, which are required to calculate the expected losses. Unfortunately, only little data are available upon to base

such an estimate [40]. One possibility is to rely on statistical reports such as [14] and [21]. Unfortunately, none of these existing data sources is without problems. Governments and security vendors have repeatedly suggested that firms under-report computer security incidents in order to avoid loss of confidence, while other observers have suggested that companies over-report the value of incidents in order to get the police interested [3].

To make things even more challenging, a successful attack's likelihood can increase through technological progress, for example by someone automating an exploit. Furthermore, the likelihood of an attack could also increase because the system's assets have become more attractive, making the security breach's impact much higher [36].

Several methods to address these challenges have been proposed [40] [29] [46].

However, when applying these models to privacy breaches, new challenges arise. First, unlike security domain, the privacy domain bears risks that are not only relevant to the affected company but rather involve further risks to the subject's privacy [19]. These risks can lead to further long term indirect costs, which comprise loss of existing customers, increased difficulty in recruiting new customers and measures to cover the brand damage [37]. These indirect costs are especially hard to quantify [19].

However, decision makers could use our results as an indirect estimation of costs that result from privacy breaches. Using this empirical data of the losses in market value associated with privacy breach announcements for similar companies as a proxy for potential loss, management might be able to make better informed decisions on security investments. Given the small magnitude of the negative abnormal returns, companies could be better suited investing their money elsewhere.

7 Conclusion

The questions of how to explore the effects and to deal with the consequences of privacy breaches are among the major issues currently discussed in the academic security community but also among public authorities and corporations. In this field, the role of privacy breach disclosure regulations and the possible incentives such regulations might provide to invest in security measures represent a matter of increasing interest. In order to explore the ambivalence of the potential effects a corresponding regulation initiative might have, we have empirically analyzed privacy breach reports and their potential impact on the market value of firms.

Our results show that privacy incidents represent an event type the capital markets do significantly react to. However, compared to other event types, the observed price reactions show an effect magnitude that question the effectiveness of corresponding regulations concerning their stimulus to invest in security measures. This especially holds true if (as we have observed) there exist business segments that appear even less affected. Our results show that more research in the field of privacy regulations is needed and that empirical analysis like ours can provide new aspects to both researchers and authorities. This also includes the development of novel approaches and mechanisms to support organizations in safeguarding their personal data stored.

References

1. Acquisti, A., Friedman, A., Telang, R.: Is There a Cost to Privacy Breaches?: An Event Study. In: Proceedings of the Twenty Seventh International Conference on Information Systems, Milwaukee, WI (2006)
2. Aharony, J., Swary, I.: Dividend and Earnings Announcements and Stockholders' Returns: An Empirical Analysis. Journal of Finance 35(1), 1–12 (1980)
3. Anderson, R., Böhme, R., Clayton, R., Moore, T.: Security Economics and the Internal Market (2008), http://www.enisa.europa.eu/doc/pdf/report_sec_econ_&_int_mark_20080131.pdf
4. Attrition.org: Data Loss Archive and Database, http://attrition.org/dataloss/ (accessed 2008-06-23)
5. Bowman, R.: Understanding and Conducting Event Studies. Journal of Business Finance and Accounting 10(4), 561–584 (1983)
6. California State Senate: Assembly Bill No. 700 (2002), http://info.sen.ca.gov/pub/01-02/bill/asm/ab_0651-0700/ab_700_bill_20020929_chaptered.pdf
7. Campbell, J.Y., Lo, A.W., MacKinlay, A.C.: The Econometrics of Financial Markets, 2nd edn. Princeton Univ. Press, Princeton (1997)
8. Campbell, K., Gordon, L.A., Loeb, M.P., Zhou, L.: The economic cost of publicly announced information security breaches: empirical evidence from the stock market. Journal of Computer Security 11(3), 431–448 (2003)
9. Carter, M., Soo, B.: The relevance of Form 8-K Reports. Journal of Accounting Research 37, 119–132 (1999)
10. Cavusoglu, H., Mishra, B., Raghunathan, S.: A Model for Evaluating IT Security Investments. Communications of the ACM 47(7), 87–92 (2004)
11. Cavusoglu, H., Mishra, B., Raghunathan, S.: The Effect of Internet Security Breach Announcements on Market Value: Capital Market Reactions for Breached Firms and Internet Security Developers. International Journal of Electronic Commerce 9(1), 69–104 (2004)
12. Chatterjee, D., Richardson, V.J., Zmud, R.W.: Examining the Shareholder Wealth Effects of Announcements of Newly Created CIO Positions. MIS Quaterly 25(1), 43–70 (2001)
13. Dehning, B., Richardson, V., Zmud, R.: The Value Relevance of Announcements of transformational Information Technology Investments. MIS Quaterly 27(4), 637–656 (2003)
14. Ernst & Young: Global Information Security Survey (2004), http://www.ey.com/global/download.nsf/International/2004_Global_Information_Security_Survey/file/2004_Global_Information Security_Survey_2004.pdf
15. Ettredge, M., Richardson, V.J.: Assessing the Risk in E-Commerce. In: Proceedings of the 35th Hawaii International Conference on System Sciences. IEEE Press, Los Alamitos (2002)
16. Fama, E.F., Fisher, L., Jensen, M.C., Roll, R.: The Adjustment of Stock Prices to New Information. International Economic Review 10(1), 1–21 (1969)
17. Fama, E.F.: Efficient Capital Markets: A review of theory and empirical work. J. Finance 25(2), 383–417 (1970)
18. Federal Trade Commission: Stipulated final judgement and order in US v Choicepoint (2006)

19. Fritsch, L., Abie, H.: Towards a Research Road Map for the Management of Privacy Risks in Information Systems. In: Alkassar, A., Siekmann, J. (eds.) Sicherheit 2008, Sicherheit Schutz und Zuverlässigkeit, pp. 1–16. Köllen Druck, Verlag GmbH, Bonn (2008)
20. Garg, A., Curtis, J., Halper, H.: Quantifying the financial impact of IT security breaches. Information Management & Computer Security 11(2), 74–83 (2003)
21. Gordon, L.A., Loeb, M.P., Lucyshyn, W., Richardson, R.: CSI/FBI Computer Crime and Security Survey, Computer Security Institute (2005), http://i.cmpnet.com/gosci/db_area/pdfs/fbi/FBI2005.pdf
22. Gordon, L.A., Loeb, M.P.: The Economics of Information Security Investment. ACM Transactions on Information and System Security 5(4), 438–457 (2002)
23. Greengard, S.: Privacy Matters. Communications of the ACM 51(9), 17–18 (2008)
24. Hovav, A., D'Arcy, J.: The impact of denial-of-service attack announcements of the market value of firms. Risk Management and Insurance Review 6(2), 97–121 (2003)
25. Hovav, A., D'Arcy, J.: The impact of virus attack announcements of the market value of firms. Information Systems Security 13(2), 32–40 (2004)
26. Im, K.S., Dow, K.E., Grover, V.: A Reexamination of IT Investment and the Market Value of the Firm: An Event Study Methodology. Information Systems Research 12(1), 103–117 (2001)
27. Ishiguro, M., Tanaka, H., Matsuura, K., Murase, I.: The Effect of Information Security Incidents on Corporate Values in the Japanese Stock Market. In: Proceedings of the Workshop on the Economics of Securing the Information Infrastructure, Washington, D.C (2006)
28. Kannan, K., Rees, J., Sridhar, S.: Reexamining the impact of information security breach announcements on firm performance. International Journal of Electronic Commerce 12(1), 69–91 (2007)
29. Lee, V.C., Shao, L.: Estimating Potential IT Security Losses: An Alternative Quantitative Approach. IEEE Security & Privacy 4(6), 44–52 (2006)
30. MacKinlay, A.C.: Event Studies in Economics and Finance. Journal of Economic Literature 35(1), 13–39 (1997)
31. Matsuura, K.: Information Security and Economics in Computer Networks: An Interdisciplinary Survey and a Proposal of Integrated Optimization of Investment. Computing in Economics and Finance (48) (2003)
32. Muntermann, J., Güttler, A.: Intraday stock price effects of ad hoc disclosures: the German case. Journal of International Financial Markets, Institutions & Money 17(1), 1–24 (2007)
33. N.N.: Garden State Briefs. The Star-Ledger, 60 (2007-06-12)
34. National Conference of State Legislatures: State Security Breach Notification Laws, http://www.ncsl.org/programs/lis/cip/priv/breachlaws.htm (accessed 2008-05-20)
35. Patell, J., Wolfson, M.: The intraday speed of adjustment of stock prices to earnings and dividend announcements. Journal of Financial Economics 13, 223–252 (1984)
36. Peeters, J., Dyson, P.: Cost-Effective Security. IEEE Security & Privacy 5(3), 85–87 (2007)
37. Ponemon Institute: Annual Study: Cost of a Data Breach: Understanding Financial Impact, Customer Turnover, and Preventative Solutions (2006), http://www.computerworld.com/pdfs/PGP_Annual_Study_PDF.pdf
38. Privacy Rights Clearinghouse: Chronology of Data Breaches, http://www.privacyrights.org/ar/ChronDataBreaches.htm (accessed 2008-06-23)

39. Roussopoulos, M., Beslay, L., Bowden, C., Finocchiaro, G., Hansen, M., Langheinrich, M., Le Grand, G., Tsakona, K.: Technology-Induced Challenges in Privacy and Data Protection in Europe (October 2008), http://enisa.europa.eu/doc/pdf/deliverables/\enisa_privacy_wg_report.pdf
40. Ryan, J.J., Ryan, D.J.: Expected benefits of information security investments. Computers & Security 25(8), 579–588 (2006)
41. Sanger, G., Peterson, J.: An Empirical Analysis of Common Stock Delistings. Journal of Financial and Quantitative Analysis 25(2), 261–272 (1990)
42. Smee, J.: Germany: Data on 17m mobile phone users stolen. The Guardian, 26 (2008-10-06)
43. Subramani, M., Walden, E.: The Impact of E-Commerce Announcements on the Market Value of Firms. Information Systems Research 12(2), 135–154 (2001)
44. Telang, R., Wattal, S.: Impact of Software Vulnerability Annoucements on the Market Value of Software Vendors: an Empirical Investigation. Carnegie Mellon University (2005)
45. Tsiakis, T., Stephanides, G.: The economic approach of information security. Computers & Security 24(2), 105–108 (2005)
46. Wang, J., Chaudhury, A., Rao, H.R.: An Extreme Value Approach to Information Technology Security Investment. In: Proceedings of the Twenty-Sixth International Conference on Information Systems (ICIS 2005), AIS, Las Vegas, Nevada, USA, pp. 347–359 (2005)
47. Weiers, R.: Introduction to Business Statistics. Thomson Brooks/Cole, Belmont (2005)
48. Wolfe, D.: Security Watch, American Banker, 172, 152, 5 (2007-08-08)
49. Wolfe, D.: Security Watch, American Banker, 172, 16, 7 (2007-01-24)

Facilitating the Adoption of Tor by Focusing on a Promising Target Group

Heiko Roßnagel[1], Jan Zibuschka[1], Lexi Pimenides[2], and Thomas Deselaers[2]

[1] Fraunhofer Institute for Industrial Engineering (IAO),
Nobelstr. 12,
70569 Stuttgart, Germany
Heiko.Rossnagel@iao.fraunhofer.de,
Jan.Zibuschka@iao.fraunhofer.de
[2] Computer Science Department,
RWTH Aachen University,
Aachen, Germany
lexi@pimenidis.org, deselaers@vision.ee.ethz.ch

Abstract. The technology for anonymous communication has been thoroughly researched. But despite the existence of several protection services, a business model for anonymous web surfing has not emerged as of today. One possibility to stimulate adoption is to facilitate it in a specific subnet. The idea is to identify a promising target group which has a substantial benefit from adopting the technology and to facilitate the adoption within that target group. We examine the feasibility of this approach for anonymity services. We identify a potential target group – consumers of pornographic online material – and empirically validate their suitability by conducting a traffic analysis. We also discuss several business models for anonymity services. We argue that providers of anonymity services should try to generate revenue from content providers like adult entertainment distributors. The latter could benefit from offering anonymous access to their products by differentiating against competitors or by selling their products at a higher price over the anonymous channel.

Keywords: Adoption, privacy, business models, anonymity services.

1 Introduction

Since Chaum proposed a method for anonymous and unobservable delivery of electronic messages [16] technologies for anonymous communication have been thoroughly researched. The concept has been adapted to internet data traffic [38], ISDN call routing [37] or mobile technology [24].

Despite their excessive needs for computational power and bandwidth, several protection services that provide anonymous communications such as Tor [21] or AN.ON [9] are offered without financial charges. However, the deployment of such technology and services has not yet reached the mass market of end users. So far only a small fraction of users are using these privacy enhancing technologies (PET) [33] and early adopters, which are necessary to reach a critical mass of adopters, have not

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 15–27, 2009.

been attracted [26]. Consequently, there is no beneficial market today for providers of anonymization services.

Also, evidence has been provided that users of anonymity services have a low willingness to pay for these services [46]. Consequently, no working business model for anonymity services has been developed so far.

While it may be argued that the intrinsic motivation of node operators to communicate anonymously themselves is enough to keep the network up and running in its initial phases, it may be doubted that the growth might continue without proper remuneration. This problem is currently seen and discussed in the Tor network [34].

In [41] several possibilities to stimulate the adoption of anonymity services have been discussed, ranging from information provisioning about the risks of unprotected online access to very invasive methods like mandatory adoption.

While most of these possibilities have to be undertaken by policy makers, some of them (i.e. bundling with complementary goods) can be applied by vendors of privacy enhancing products. One such approach is to facilitate the adoption in a specific subnet. The idea is to identify a promising target group which has a substantial benefit from adopting the technology and to facilitate the adoption within that target group. If this group is large enough, demographically spread and well coordinated this could lead to natural adoption [36]. Anonymity services could therefore be selectively offered to users of services which have a more obvious demand for anonymity and a high willingness to pay for them [41].

In this contribution we examine the feasibility of this approach. We first identify a potential target group – consumers of pornographic online material – and then empirically validate their suitability for facilitating adoption of anonymity services, by conducting a traffic analysis. We also discuss different business models and possible approaches for revenue generation.

2 Related Work

In recent years there has been a growing body of research on the economics of privacy and privacy enhancing technologies. Most of the work is concerned with the determination or measurement of the value of privacy for individuals [47]. Huberman et al. [30] used reverse-prize auctions to identify the monetary value of private information to individuals. Their results show that a trait's desirability in relation to the group impacts the amount people demand for revealing this information. For example individuals weighing slightly below average required little compensation to publicize this fact. In contrast, those who weighed more and might therefore fear embarrassment or stigmatization demanded relatively high compensation [30].

Other studies examined the circumstances under which users are willing to disclose private information [3] [1] [8]. Their results show that individuals who genuinely would like to protect their privacy may not do so because of psychological distortions such as hyperbolic discounting, under insurance, self-control problems and immediate gratification [3] [1]. This demonstrates discrepancies between attitudes towards privacy and actual behaviour [8].

The results of [7] show that perceptions of a web merchant's trustworthiness can be high even when privacy and security features are weak. As a possible explanation

for this result they suggest that the respondents may have lacked familiarity with or understanding of the seals and statements.

In more market oriented research, Acquisti argues that the market of privacy conscious individuals willing to pay for their protection is small and therefore will not be satisfied [2]. He attributes the small market size to the low amount of people who are conscious of their security needs [1] and willing to pay for it. Furthermore, he claims that "while actual usage costs of privacy enhancing technologies are low once adopted, their adoption fees are high because they involve significant switching costs" [2]. Shostack [43] argues that when privacy protection is offered in a clear comprehensible way it does sell well. He supports this argument with several examples such as draperies and curtains. Accordingly he concludes that complex technologies that protect against nebulous threats will not sell well. Feigenbaum et al. [26] attribute the missing adoption of privacy-technology to economic factors like network externalities, asymmetric information, and moral hazard.

The willingness of users to pay for anonymization services has been researched in [45] and [46], in which the results of a survey of over 5000 users of the AN.ON privacy service were presented. When asked about their willingness to pay for anonymity services, 40% of the participants – all of them already users of an anonymity service – were not willing to pay anything at all. Approximately 50% were willing to pay between €2.50 and €5.00 per month while 10% would have paid more then €5.00.

There have also been observatory approaches, relying on the classification of logged traffic into several categories [25]. However, these results seem to be somewhat contradictory, concerning both background of usage (with self-reports overstating professional use compared with the measurement/categorization approach) and use cases (understating pornographic material). While this discrepancy may be explained with the well-documented bias of people to overstate their privacy sensitivity [3] [47] or the generally weak validity of self-report studies in the context of sexuality [10], to our knowledge, the publications based on direct measurements of Tor traffic do not describe a clear methodology that would allow us to retrace how the results were obtained. To overcome these restrictions of the available material, we decided to do a new analysis for this paper, with a clearly documented methodology described in Section 4.

3 Identifying Potential Target Group

In order to facilitate adoption of anonymity services in a subnet, a promising target group has to be identified. Ideally, members of this target group should have a high demand for anonymity and a high level of innovativeness. Furthermore, since the focus is on commercial anonymity services, users should also have a high willingness to pay for anonymity services or associated products and services.

One such target group could be consumers of pornographic online material. Obviously they have a high demand for anonymity [19]. For example, the German video on demand service provider videoload.de is running a commercial in which the absence of embarrassing moments (i.e. when returning pornographic tapes) is advertised as one of the key features of their service[1] [50].

[1] Please note, however, that videoload.de does not provide anonymous access to their services.

Also, users of pornographic material have shown a high level of innovativeness in the past [18]. Adult-oriented content has been reported as a main driver for the establishment of several wide-spread technologies and infrastructures, such as VCRs, DVDs, computers, or the internet itself [19] [18]. Even the success of VHS over the Betamax VCR standard, has been attributed to pornography [5]. In addition, consumers of pornography have shown a high willingness to pay in the past [18] [4].

Furthermore, the market for pornographic material is quite substantial. In Italy alone, the turnover deriving from selling or renting pornographic movies to the final users is about 224,000,000 € [23]. Globally, pornography generated a turnover of over $ 97 Billion in 2006 with over $ 2.8 Billion coming from internet pornography [48].

Also, it is notable that users are obviously consuming a large amount of multimedia content over the Tor network, in spite of its reported sluggishness [22]. So, while fast response times are a big factor when browsing web sites [29], for larger multimedia content, this factor seems less deterring, especially in the case of adult entertainment.

Looking at the numbers given in [25], it is quite obvious that this is a promising use case for web anonymizers. However, tapping into the potential of the pornography market should be done very cautiously, as mixing adult content and mainstream media is not acceptable in many cultures [32] [6]. So, to be successful in this context, the service needs to be carefully aimed at the proper audience, and make sure not to mix the mainstream and adult-oriented contexts.

If consumers of pornographic online material have a higher demand for anonymity and a higher level of innovativeness than regular internet users, the percentage of user's accessing pornographic material should be higher within the Tor Network than on the internet. This should also be visible in the percentage of pornographic material within the data traffic transferred over these networks. Therefore, we formulate a research hypothesis:

***H1**: Traffic transferred over the Tor Network contains more pornographic material than regular internet traffic.*

4 Research Methodology

In order to test our hypothesis we analysed the output[2] of a Tor exit node and compared these results to those of an outgoing proxy of a German university. The choice of a university proxy as a control group to Tor was based on the fact that we were unable to get a more representative data set from an internet service provider (ISP), because the data we would have needed is usually not recorded. Similarly, we investigated a single Tor exit node as a representative of the entire network, as such data is not recorded by many exit nodes. We hold that our data is of interest as traffic statistics of this granularity are usually not publicly available.

We restricted our analysis to HTTP traffic for several reasons. HTTP is, after Peer to Peer (P2P), the second largest traffic class [42]. While P2P traffic comprises a significant amount of data traffic, its analysis is tedious as the content is often

[2] Due to the nature of Tor, plain HTTP requests and their responses can only be seen when leaving the network; their origin is, however, untraceable.

transferred in chunks and from different hosts. This makes an analysis of the type of transferred data very difficult, if not impossible. Also, the ratio of the transferred bytes per connection suggests that P2P is mostly used for videos rather than still pictures [33].

Furthermore, HTTP has a long tradition of scientific analysis and there are well known methods for analysing HTTP traffic in a privacy-preserving way. In order to stick to lawful research[3] we used the following method of data acquisition and analysis, which was developed together with data protection officers. First, we collected a set of URLs which have been requested by users over either Tor or the university's proxy. We did not save any information except the requested URL itself; this includes any additional header information like Cookies, etc. Requests sent with methods other than GET, e.g. POST requests, or any requests containing GET-variables, were completely left out of the analysis. We also blurred the time stamp of a request to store only the year and the month and then reordered the requests per month so to hide any information about the order of requests and also thwart guesses narrowing down the date of a request within a month. In the case of Tor, we also could not collect the source IP of the request by definition, in the case of the university's proxy we asked the proxy's administrator specifically not to copy the IPs, even if they were recorded. All of this was done automatically and any data not of interest was already discarded at this stage. Then, more people were involved to select a subset of the URLs. These were polled by a program and the content was dispatched without these helpers being able to store it. Finally, the content was analysed by people not knowing its source. This procedure ensured that the researchers for the actual analysis did not have access to the original data, whereas the others did not know the content of the URLs[4].

In order to reliably classify the type of content we had to inspect the type of the actually transferred data. This holds true as especially in the case of free image web hosting it is not possible to learn the type of the image by the URL's pattern alone. To make sure, we retrieved 7,000 images from each of the sets of URLs with the procedure described above. To rule out that the results would be drowning in a vast amount of small pictures that are used as icons or frame borders in web pages, we limited the search to images with a size of larger than 10,000 bytes.

We then used a set of pattern matching techniques described in [20], to classify the images into five categories:

- Class 0 definitely inoffensive images.
- Class 1 lightly dressed persons.
- Class 2 partly nude persons.
- Class 3 nude persons.
- Class 4 pornographic, i.e. one or more persons engaging in sexual intercourse.

In order to minimize the error of the classification process, we used the automated classification method for a preliminary result, as the pictures extracted from the URL streams were to diverse in order to produce acceptable results in the first run: the

[3] The authors of the aforementioned [33] were threatened to be legally prosecuted due to their methodology.

[4] Also note that this procedure preserved picture owner's privacy which use any kind of authentication mechanism to prevent access to their pictures.

overall correct classification rate was 33% in the case of the pictures coming from Tor and 44% for the pictures from the university's proxy. The classification ratio was raised to 70%, resp. 75% if a deviation of one class was deemed an acceptable error.

5 Empirical Results

The input of the automated classification was enhanced in a manual process of re-classification. Due to time constraints and data protection concerns we only used a random sample of 1,000 images from each set for a final manual classification. The results of our analysis are listed in Table 1. Content categorized by image recognition technique

Table 1. Content categorized by image recognition technique

	Class 0	Class 1	Class 2	Class 3	Class 4
Tor	28%	15%	15%	14%	28%
Plain	66%	7%	8%	8%	11%

Our results show that the Tor network has a much higher percentage of sexually oriented material than normal traffic: 72% of the pictures in the Tor network and 34% in normal traffic were classified in Class1 to Class 4. Even if material from “Class 1", which can be encountered in everyday's advertising in most western countries, is not counted, the percentage remains substantially higher for the Tor network (57% vs. 27%). While the percentages for the Classes 1-3 are about twice as high for Tor traffic, this difference spreads even further for Class 4 (28% vs. 11%).

To test our hypothesis, we used a χ^2-test to check if both the differences between these distributions are due to statistical fluctuation or whether they differ significantly. The test clearly shows that the distributions differ significantly with $p<0.0001$. We also used the one-sided Wilcoxon Rank Sum test to check whether the traffic from Tor contains significantly more pornographic material than plain traffic. The obtained result confirmed our hypothesis with $p<2.2 * 10^{16}$, which means that the Tor traffic *does* contain a significant higher percentage of adult traffic.

This correlates with results of related work that activities related to sexual behaviour are very privacy sensitive and therefore subject to privacy protection techniques [49].

6 Business Models

Having identified a potential target group in which to facilitate adoption and shown its presence in the targeted medium, we present possible business models and analyse their motivation, restrictions, and opportunities.

6.1 User Side Revenue Generation

The only approach that has been applied so far is to directly offer anonymity services to potential users. In this scenario the users have to install software on their computers

in order to anonymize the traffic. Examples of this approach are JonDoFox [31], which combines Firefox with Jondos, a commercial spin-off of the AN.ON project and the XeroBank Browser [44], which is a modification of the popular Firefox browser, readily compiled with a client for the Tor network.

This approach has not been successful even for services that are free of charge [41]. Potential adopters are neither aware of the privacy risks they face when communicating online nor of the available technology to protect themselves against these risks [41]. Also, these PETs are preventive innovations, which are ideas that are adopted by an individual at one point in time in order to lower the probability that some future unwanted event will occur [40]. Preventive innovations usually have a very slow rate of adoption, because the unwanted event might not happen even without the adoption of the innovation. Therefore, the relative advantage is not very clear cut [40]. Furthermore, users of anonymity services have been shown to have a small willingness to pay in the past [46].

Due to the complexity and missing user-friendliness of current anonymity solutions, users also endure a large amount of search costs in order to setup the software correctly at their side. The technical problems include possible information leakages due to DNS [21], browser plugins [27], or plain configuration faults at the installation [21] [9]. It is widely agreed that the correct installation and usage of local proxies and related browser configuration is often too hard to be properly carried out by users [17].

Some of these problems, like for example the willingness to pay, might be reduced if consumers of pornographic material are specifically targeted. Nevertheless, a success of this model remains rather doubtful.

6.2 Advertising at the Exit Node

The second possibility that we discuss is the generation of additional revenue by implementing advertising at the exit node. This is the last node of the anonymity network, where the user's traffic leaves the network and is finally redirected to its destination. This fact makes the operator of the last node one of the most vulnerable parts in the setup of any overlay network. If a request which is forwarded out of the network contains or requests any malicious content, it is this operator that is made responsible for this in the first place. For this reason, only a minority of node operators are actually allowing traffic to exit through their nodes out of the overlay network. Due to this fact, it is critical for any network to have an appropriate number of exit nodes. Therefore, it can be argued that remunerating these operators more than others, or even remunerating only exit node operators, can be accepted from an ethical or fairness point of view. Additionally, a solution propagating the revenues from the exit node back across the anonymization network may be implemented, so the discussion of the exit nodes may serve as a basis for progress in recuperation of other nodes.

In contradiction to the wide-spread, but wrong, believe of average users that all traffic through anonymity networks cannot be seen by any third parties, the content of the actual request is plainly visible to the exit node. Thus, the exit node operator can deploy a software component to rewrite the user's request as well as the related content before forwarding them to their respective targets. Rewriting can include the following parts:

(1) The node operator may replace existing advertisements in web pages with advertisements where the exit node operator profits from possible click rates or sells. Although the exit node operator can replace arbitrary content, for an economical gain, it would be sufficient to replace ads, advertisement, and banners in the transferred pages. With this technique, the exit node operators could profit from the flow of traffic through their node: if they replace a certain amount of advertisement with a set of links to their customers, they get remunerated for their services.
(2) Another rather aggressive way would be to redirect users sending queries to search engines directly to partner sides of the exit node operator, or just reorder the search engine's result.
(3) Finally, the exit node operator is capable to inject small pieces of Javascript into an HTML page in order to trigger loading of pop-up windows. As we personally feel that pop-ups are becoming a growing annoyance, we would not recommend this technique and only list it for completeness purposes.

It can be said with some confidence that such a system should be able to recover costs of operation. Assuming a cost per Gigabyte of about US$0.20, an average click rate on web advertisement of around 2 to 3 per thousand advertisements [14], and a financial gain of US$0.01 per click, there should be at least 20 clicks per Gigabyte, i.e. 8,000 advertisement displayments.

This means that if an exit node operator owns one banner per 131 Kilobyte of traffic, he can do a break-even. As has been shown in [39], the size of an average HTML-page is 5 Kilobyte. Even if we multiply it by a safety factor of 10, resulting in an average HTML-page size of 50 Kilobyte, it would be sufficient for an exit node operator to only insert or replace an advertisement in about every third page.

Some of these methods presented here may face legal problems in some jurisdictions, because they are defrauding the content providers, owners of the underlying web sites, e.g. based on intellectual property regulation. Besides legal risks, the exit node operators also face a second problem: the Tor community itself is known to be highly critical of any kind of content modification by the exit node operators. If a certain node starts to actively replace content in the relayed data streams, it might face exclusion from the Tor network.

So, while the numbers add up for this method, it faces serious legal problems, and motivates investigation of business models involving the service and content providers in a more prudent fashion.

6.3 Revenue Generation from Content Providers

Another possible approach is revenue generation at the content providers whose products the users consume over the anonymization network. There may be several motivations of revenue generation in this context:

(1) The content provider may be able to charge his users a premium for services and products offered to them anonymously, which should result in an additional payoff if a willingness to pay for privacy exists on the user side [12].
(2) The content provider may have a willingness to make payments assuring the continuous operation of a third party anonymization network, if it is used by a large set of his customers in conjunction with its services.

(3) Providing anonymous access to its products could enable the content provider to differentiate itself against competitors.

As our traffic analysis has shown, a suitably large number of users are consuming online pornography. So, adult content providers seem to be operating in a niche market where an increased user's need for privacy exists. Given that these user preferences translate into an increased willingness to pay (a sensible assumption, given that the public Tor network we investigated has significant performance issues [22]), an investment of Tor service providers should result in an ability to price their products at a premium. Böhme and Koble [12] point out that this assumption about privacy-enhancing technologies holds under a broad set of assumptions. The results of [4] show that especially non-Internet savvy individuals whose time is valuable are willing to pay for online pornography in order to avoid hidden costs (e.g. search costs or the risk of virus infection).

This suggests a business model in which adult service providers cooperate with a subset of nodes in an anonymization network, resulting in their ability to offer anonymized services. This subset of nodes may consist of Tor exist nodes, assuring recompensation for their increased exposure, or coordinated MIX cascades, which may be able to offer special services like increased performance to the customers of the service provider. This may also be possible using Tor, however, and both systems have their strengths and weaknesses [11]. However, for demonstrating the technological viability of this business model, cooperation with a commercial anonymization cascade operator seems like the more traceable choice.

There are several possibilities for the implementation of such a system. A solution that blocks all incoming traffic except the one from a certain set of anonymization exit nodes has been proposed in [28]. Also, systems, that implement full client-side anonymisation (including e.g. a streaming media player with built-in anonymization), or full server-based anonymization are both viable for executing this business model.

As can be seen in comparison with the business models discussed in the previous sections, this offers several key advantages. It is easy to operate this system in a lawful way for both node operators and end users. Service providers can realize additional revenue streams by leveraging their users' willingness to pay for privacy.

However, to make this business model work, it is necessary for anonymization node operators to establish relations with potentially interested content providers. Using the results of our earlier traffic analysis, we have identified adult entertainment as a viable application field.

One essential prerequisite for implementing this business model is the establishment of a working model for anonymous payment at the content provider. Anonymous payment can be achieved by using highly sophisticated cryptographic solutions such as [15] or [13]. However, these payment systems have not been applied in real business scenarios so far and exist mainly as research prototypes. Also their chances of success have been questioned [35]. An alternative solution would be the use of pre-paid cards, such as calling cards or anonymous pre-paid SIM cards. Another possibility would be to accept cash payments send with regular mail on behalf of specific pseudonymous user accounts, which is one of the methods used by JonDos [31].

7 Limitations

We have shown that the percentage of pornographic material is higher in the Tor-Network than in plain Internet traffic, and we concluded that the consumers of that material have a higher demand for anonymity services. However, a higher willingness of these consumers to pay for anonymous access to online pornography has not been empirically validated. We encourage future research in that direction.

For anonymity service provider to successfully generate revenue from adult content providers, a working method of anonymous payment has to be established. So far such a method is missing. We have only sketched several possible approaches. Furthermore, a cost-benefit analysis for content provider should be performed in the future, in order to support our argument of the feasibility of that approach.

8 Conclusion

While the technology for anonymous communication has been thoroughly researched and despite the existence of several protection services, a business model for anonymous web surfing has not emerged as of yet. So far only innovators are using privacy enhancing technology and early adopters, which are necessary to reach a critical mass of adopters, have not been attracted. Therefore, there is no beneficial market today for providers of anonymity services. With our contribution we tried to bridge this gap. In order to find a suitable target group for anonymity services we conducted a traffic analysis. Our results show that there is a significant demand for anonymous access to pornographic material. Furthermore, the users of pornographic material have shown a high level of innovativeness and a high willingness to pay in the past and have been drivers of technological innovation. Therefore, it seems to be the logical choice to target this specific customer group for the deployment of anonymous services. We then examined the different possibilities to generate revenue for such anonymous services. We concluded that business models that create revenue by directly offering anonymity service to users have failed in the past. Models that aim at creating revenue by inserting advertisements of third parties into the traffic at the exit node might be profitable but face serious legal problems. Finally, we argued that the most promising approach is to generate revenue from content providers like adult entertainment distributors. The latter could benefit from offering anonymous access to their products by differentiating against competitors or by selling their products at a higher price over the anonymous channel.

References

[1] Acquisti, A., Grossklags, J.: Privacy and Rationality in Individual Decision Making. IEEE Security & Privacy 3(1), 26–33 (2005)
[2] Acquisti, A.: Privacy and Security of Personal Information: Economic Incentive and Technological Solutions. In: Camp, J., Lewis, R. (eds.) The Economics of Information Security, pp. 1–9. Kluwer, Dordrecht (2004)
[3] Acquisti, A.: Privacy in Electronic Commerce and the Economics of Immediate Gratification. In: Proceedings of the EC 2004. ACM, New York (2004)

[4] Ang, E.X.Y., Kwan, J.W.Y., Teo, J., Chua, C.E.H.: Why Do People Pay for Information Goods?: A Study of the Online Porn Industry. In: Proceedings of the Twelfth Americas Conference on Information Systems (AMCIS 2006), AIS, Acapulco, Mexico, pp. 73–77 (2006)
[5] Angell, I.O.: Ethics and Morality: a business opportunity for the Amoral? Journal of Information System Security 3(1), 3–18 (2007)
[6] Bazak, A., King, S.A.: The Two Faces of the Internet: Introduction to the Special Issue on the Internet and Sexuality. CyberPsychology & Behavior 3(4), 517–520 (2000)
[7] Belanger, F., Hiller, J.S., Smith, W.J.: Trustworthiness in electronic commerce: the role of privacy, security, and site attributes. Journal of Strategic Information Systems (11), 245–270 (2002)
[8] Berendt, B., Günther, O., Spiekermann, S.: Privacy in E-Commerce: Stated Preferences vs. Actual Behavior. Communications of the ACM 48(4), 101–106 (2005)
[9] Berthold, O., Federrath, H., Köpsell, S.: Web MIXes: A system for anonymous and unobservable Internet access. In: Federrath, H. (ed.) Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability, pp. 115–129. Springer, Heidelberg (2000)
[10] Brener, N.D., Billy, J.O.G., Grady, W.R.: Assessment of Factors Affecting the Validity of Self-Reported Health-Risk Behavior Among Adolescents: Evidence From the Scientific Literature. Journal of Adolescent Health 33, 436–457 (2003)
[11] Böhme, R., Danezis, G., Díaz, C., Köpsell, S., Pfitzmann, A.: Mix Cascades vs. Peer-to-Peer: Is One Concept Superior. In: Martin, D., Serjantov, A. (eds.) PET 2004. LNCS, vol. 3424, pp. 243–255. Springer, Heidelberg (2005)
[12] Böhme, R., Koble, S.: Pricing Strategies in Electronic Marketplaces with Privacy-Enhancing Technologies. Wirtschaftsinformatik 49(1), 16–25 (2007)
[13] Camenisch, J., Piveteau, J., Stadler, M.: An Efficient Fair Payment System. In: Proceedings of the 3rd ACM Conference on Computer and Communication Security (CCS 1996), New Dehli, India (1996)
[14] Chatterjee, P., Hoffman, D.L., Novak, T.P.: Modeling the clickstream: Implications for web-based advertising efforts. Marketing Science 12(4), 520–541 (2003)
[15] Chaum, D.: Blind Signatures for Untraceable Payments. In: Crypto 1982, pp. 199–203. Plenum, New York (1983)
[16] Chaum, D.L.: Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. Communications of the ACM 24(2), 84–88 (1981)
[17] Clark, J., Van Oorschot, P.C., Adams, C.: Usability of anonymous web browsing: an examination of tor interfaces and deployability. In: Proceedings of the 3rd symposium on Usable privacy and security (SOUPS 2007), New York, NY, pp. 41–51 (2007)
[18] Coopersmith, J.: Pornography. In: Technology and Progress, ICON, vol. 4, pp. 94–125 (1998)
[19] Cronin, B., Davenport, E.: E-rogenous Zones: Positioning Pornography in the digital Economy. The Information Society 17(1) (2001)
[20] Deselaers, T., Keysers, D., Ney, H.: Discriminative Training for Object Recognition using Image Patches. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005), San Diego, CA, USA, pp. 157–162 (2005)
[21] Dingledine, R., Mathewson, N., Syverson, P.: Tor: The Second-Generation Onion Router. In: Proceedings of the 13th USENIX Security Symposium, San Diego, pp. 303–320 (2004)
[22] Dingledine, R., Mathewson, N.: Anonymity loves company: Usability and the network effect. In: The Fifth Workshop on the Economics of Information Security (WEIS 2006), University of Cambridge (2006)
[23] D'Orlando, F.: The Market for Pornography in Italy: Empirical Data and Theoretical Considerations (2008-04-25), http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1125129

[24] Federrath, H., Jerichow, A., Kesdogan, D., Pfitzmann, A., Spaniol, O.: Mobilkommunikation ohne Bewegungsprofile. In: Pfitzmann, A., Müller, G. (eds.) Mehrseitige Sicherheit in der Kommunikationstechnik, pp. 169–180. Addison Wesley, Boston (1997)
[25] Federrath, H.: Privacy Enhanced Technologies: Methods - Markets - Misuse. In: Katsikas, S.K., López, J., Pernul, G. (eds.) TrustBus 2005. LNCS, vol. 3592, pp. 1–9. Springer, Heidelberg (2005)
[26] Feigenbaum, J., Freedman, M., Sander, T., Shostack, A.: Economic barriers to the deployment of existing privacy technology. In: Proceedings of the Workshop on Economics and Information Security, Berkley, CA (2002)
[27] FortConsult: Practical onion hacking: Finding the real address of tor clients (2006)
[28] Fritsch, L., Roßnagel, H., Schwenke, M., Stadler, T.: Die Pflicht zum Angebot anonym nutzbarer Dienste: Eine technische und rechtliche Zumutbarkeitsbetrachtung. Datenschutz und Datensicherheit (DuD) 29(10), 592–596 (2005)
[29] Galletta, D.F., Henry, R., McCoy, S., Polak, P.: Web site delays: How tolerant are users? Journal of the Association for Information Systems 5(1), 1–28 (2004)
[30] Huberman, B.A., Adar, E., Fine, L.R.: Valuating Privacy. IEEE Security & Privacy 3(5), 22–25 (2005)
[31] JonDos GmbH: JonDoFox: Private and Secure Web Browsing (2.1.2), http://www.jondos.de/en/jondofox (accessed 2008-11-30)
[32] Lambiase, J.: Sex: Online and in Internet Advertising. Lawrence Erlbaum Associates, Mahwah (2003)
[33] McCoy, D., Bauer, K., Grunwald, D., Kohno, T., Sicker, D.C.: Shining light in dark places: Understanding the tor network. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 63–76. Springer, Heidelberg (2008)
[34] Murdoch, S.J.: Economics of Tor performance (2007-07-18), http://www.lightbluetouchpaper.org/2007/07/18/economics-of-tor-performance/
[35] Odlyzko, A.M.: The Case Against Micropayments. In: Wright, R.N. (ed.) FC 2003. LNCS, vol. 2742, pp. 77–83. Springer, Heidelberg (2003)
[36] Ozment, A., Schechter, S.E.: Bootstrapping the Adoption of Internet Security Protocols. In: Proceedings of the Fifth Workshop on the Economics of Information Security (WEIS 2006), Cambridge (2006)
[37] Pfitzmann, A., Pfitzmann, B., Waidner, M.: ISDN-mixes: Untraceable communication with very small bandwith overhead. In: Proceedings of the GI/ITG Conference on Communication in Distributed Systems, pp. 451–463 (1991)
[38] Pfitzmann, A., Waidner, M.: Networks without User Observability – Design Options. In: Pichler, F. (ed.) EUROCRYPT 1985. LNCS, vol. 219, pp. 245–253. Springer, Heidelberg (1986)
[39] Pitkow, J.E.: Summary of WWW characterizations. Computer Networks and ISDN Systems 30(1-7), 551–558 (1998)
[40] Rogers, E.M.: Diffusion of Innovations, vol. 5. Free Press, New York (2003)
[41] Roßnagel, H.: Bootstrapping the Adoption of Privacy Enhancing Technology. In: Proceedings of the 1st IDIS Workshop, Arona, Italy (2008)
[42] Schulze, H., Mochalski, K.: Internet Study 2007: The Impact of P2P File Sharing, Voice over IP, Skype, Joost, Instant Messaging, One-Click Hosting and Media Streaming such as YouTube on the Internet (2007), http://www.ipoque.com/resources/internet-studies/internet-study-2007
[43] Shostack, A.: People Won't Pay For Privacy. In: Reconsidered, 2nd Annual Workshop Economics and Information Security, University of Maryland (2003)
[44] Softonic: XeroBank Browser, http://xerobank-browser.softonic.de/ (accessed 2008-05-20)

[45] Spiekermann, S.: Die Konsumenten der Anonymität: Wer nutzt Anonymisierungsdienste? Datenschutz und Datensicherheit (DuD) 27(3), 150–154 (2003)
[46] Spiekermann, S.: The desire for privacy: Insights into the views and nature of the early adopters of privacy services. International Journal of Technology and Human Interaction 1(1) (2004)
[47] Syverson, P.: The Paradoxical Value of Privacy. In: 2nd Annual Workshop Economics and Information Security, University of Maryland (2003)
[48] Top Ten Reviews: Internet Pornography Statistics (2009), http://internet-filter-review.toptenreviews.com/internet-pornography-statistics.html#anchor1
[49] Tsai, J., Egelman, S., Cranor, L., Acquisti, A.: The Effect of Online Privacy Information on Purchasing Behavior: An Experimental Study. In: Proceedings of Workshop on the Economics of Information Security, Pittsburgh, PA (2007)
[50] Videoload.de: Pastewka Videoload Peinliche Momente, http://www.youtube.com/watch?v=5rBK4AUljUg (accessed 2008-04-25)

A Parallelism-Based Approach to Network Anonymization

Igor Margasiński

Institute of Telecommunications, Warsaw University of Technology
igor@tele.pw.edu.pl

Abstract. Considering topologies of anonymous networks we used to organizing anonymous communications into hard to trace paths, composed of several middleman nodes, towards hiding communication's origin or destination. In anonymity the company is crucial. However, the serial transportation of content imposes a costly tradeoff between speed of communication and a level of privacy.

This paper introduces a framework of a novel architecture for anonymous networks that hides initiators of communications by parallelization of anonymous links. The new approach, which is based on the grounds of the anonymous P2P network called P2Priv, does not require content forwarding via a chain of proxy nodes to assure high degree of anonymity. Contrary to P2Priv, the new architecture can be suited to anonymization of various network communications, including anonymous access to distributed as well as client-server services. In particular, it can be considered as an anonymization platform for these network applications where both privacy and low delays are required.

Keywords: Communication system security, privacy, anonymity, anonymous network architectures.

1 Introduction

Anonymous communications by means of public packet networks involve two contradictory tasks: the first is transport and routing—which requires a detail information on origin and destination points of communications, and the second is anonymization—which is basically aimed at hiding of this information and especially an association between them. Certainly, addressing information is essential for a successful delivery of content and therefore it cannot be expunged. Hence in general, the ally of anonymous communication is collective [1,2,3]. The single word *crowd* speaks volumes about anonymity. The more numerous a set of actors involved in an information exchange, *blending into a crowd* is easier to achieve. And then, the higher traffic volume among these actors, the faster our traffic can be hidden and exchanged. In general, to represent such a collective, an operation of anonymous networks is based on a sequential traffic forwarding by a subgroup of network nodes, also known as *proxy chaining* [4]. However, substantial delays mount up in this way. This paper introduces an alternate architecture

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 28–43, 2009.

for anonymous networks, a network privacy preserving parallel topology, where network actors organize themselves in parallel links. The topology of the new solution evolves from the anonymous P2P network called P2Priv [5]. However, the applications of the new architecture are not limited to P2P content distribution and its deployment can be considered for generic-purpose anonymous networks.

The rest of the paper is organized as follows. Section 2 provides an introduction to topology issues of anonymous networks. Section 3 describes a model of an adversary while Section 4 contains an anonymity analysis of parallel architecture of P2Priv applied first, in accordance with its intended use, for P2P content distribution, and secondly, for client-server like scenarios. Section 5 contains a description of our novel solution able to assure anonymity for centralized network services, while its anonymity analysis is presented in Section 6. Conclusions and discussion of a future work are included in Section 7.

2 Background

The topology of anonymous networks has been widely discussed since the introduction of Chaum's Mix-net anonymous network [6] and among a variety of anonymous networks, the most attention has been devoted to the interconnection issues of Mix-based nodes. Originally, the route through a cascade of Mixes was fixed. Further improvements allowed a sender to randomly select a path for each message in the so called free-route topologies [7,8]. Hybrid models with a restricted number of connections and a path selection narrowed to restricted-routes were proposed in [9]. Both fixed cascades and fully interconnected Mix networks with random routes have various constraints and the advantage of each depends primarily on the area of their deployment and the scale of the network [7,10,11,12].

Besides Mixes, other designs of anonymous networks were proposed within individual interconnection strategies. Anonymous message-by-message routing called *onion routing* was proposed by Goldschlag et al. [4]. Today, several implementations of this concept are available, including the low-latency network called *the second generation onion router* (TOR) designed by Dingledine et al. [13]—the general purpose anonymous network of the world-wide range.

Reiter et al. proposed other low-latency anonymous network called *Crowds* with anonymous routing based on the *random-walk* step [14]. In this strategy senders do not influence a path selection which is determined in a random manner in each hop sequentially. Both hop-by-hop and message-by-message routing strategies are less robust against traffic dropping by proxy nodes than Mix-network. Still, their simplicity makes them attractive in practice [15].

By and large, the common feature of anonymous networks is their **serial** architecture and a formation of untraceable paths via middle-man nodes for anonymous content forwarding. Due to such praxis, and in combination with traffic encryption and anonymization techniques deployed inside proxy nodes (e.g., content batching, aggregations, and permutations), an observer of the anonymous networks cannot practically point out senders and receivers in a batch of inter-mixed content flows via middle-man nodes.

2.1 Parallel Architecture of P2Priv

The serial content forwarding, known from today's anonymous networks, was omitted in the architecture of the anonymous peer-to-peer overlay network called P2Priv (peer-to-peer direct and anonymous distribution overlay) proposed by Margasiński et al. in [5]. The topology of the P2Priv network, in respect to content transportation, involves a number of additional virtual links similar to classical anonymous networks; however, it is arranged in **parallel** manner. Figure 1 illustrates the parallel architecture of P2Priv with solid lines representing plain-text communication and dotted lines corresponding to communication secured by the anonymization techniques. Let us have a closer look at the P2Priv architecture. Before a content transportation, a signalization token with metadata describing the content is forwarded over classical anonymous paths towards formation of so called *cloning cascades* (*CC*). The well-known anonymous techniques, i.e., Mix network and random walk algorithm, are utilized in hiding the initiator of the *CC* and in the anonymization process of this *lightweight* communication. Note that traffic comprised by numerous and short messages sent by random nodes is in favour of a Mix-net performance. Then, after a random interval of time, each *CC* member, i.e., group of the so called *clones* and the true initiator, communicates directly and independently with a destination node or nodes towards content transportation. The anonymity of P2Priv is based on a collective formed in a parallelism-based manner, as every content exchange in P2Priv is accompanied by a simultaneously generated exchange of the same content initiated by random nodes. A process of finding the true initiator among P2Priv nodes is hard to perform even for an adversary able to collude a significant range of network nodes.

The results of anonymity and traffic performance analysis are promising for P2Priv [5,16]. However, they have been obtained for a distributed environment which is not always available in general-purpose communications.

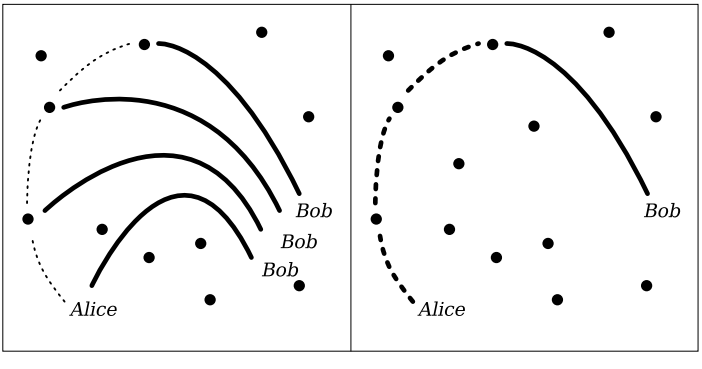


Fig. 1. P2Priv architecture (*left*) as compared to classical anonymous network (*right*)

3 Threat Model

We consider a threat model with an adversary who controls a colluding fraction ρ of all network nodes N. We assume that these malicious nodes are able to conduct both passive and active attacks. We consider a static adversary unable to arbitrarily adapt the set of malicious nodes among user nodes and we assume that colluding nodes are uniformly distributed among equivalent nodes. However, whenever we will deal with server-like nodes or just centralized points of communication, we will treat these nodes as easy to choose points of observation and assume that these nodes can be compromised. Using information entropy measures and based on the information theoretic anonymity measurement model [17,18], an adversary who posses no information about the system can describe his/her uncertainty in successful finding of the initiator of a particular action (let us call her *Alice*) as

$$\mathcal{H}_{\max} = -\sum_{i=1}^{|N|(1-\rho)} \frac{1}{|N|(1-\rho)} \log_2\left(\frac{1}{|N|(1-\rho)}\right) = \\ -\log_2(|N|(1-\rho)). \tag{1}$$

4 Anonymity Analysis for P2Priv

As the new architecture proposed in this paper share some primitives with P2Priv, we will apply the threat model to P2Priv architecture first and discuss its anonymity in two service scenarios. In the first one, likely to occur in fully distributed P2P systems, we will consider content distribution (e.g., P2P file-sharing) among equivalent peers. Secondly, we will analyze anonymity offered by P2Priv for typical client-server network services where client nodes connect repeatedly to some server (e.g., an access to a popular Web server). Note that models, similar to the first scenario, has been investigated previously in [5,19,16] with different adversary possibilities considered. The second scenario has not been considered yet. For simplicity purposes, the scenarios will be referred to as **P2P Scenario** and **Client-Server Scenario** respectively.

The detailed description of P2Priv was introduced in [16]. At a glance, cloning cascade (CC) of P2Priv is established in random-walk manner with a mean length described by probability p_f, as follows. *Alice*, who is the first member of the CC, generates and sends a token with a content id to a randomly selected node from N. This random node, chosen to be the next CC member, flips an asymmetric coin to decide whether to forward the receiver token, with probability p_f, to another random node (also, compare it with the random-walk step defined in the Crowds network [14]). This step repeats until finally one of the nodes receiving token decides (with probability $1 - p_f$) to terminate the token's forwarding process:

$$|CC| = \sum_{i=2}^{\infty} i\, p_f^{i-2}(1-p_f) = \frac{p_f - 2}{p_f - 1}. \tag{2}$$

Each step of the token's random-walk is sent by a Mix-network layer formed for this purpose by user nodes (N). After establishing CC, randomly delayed direct connections originate from CC nodes towards content transportation (Figure 1). The analysis of various attacks on P2Priv, presented in [19], shows that a secure configuration of P2Priv starts from $p_f = 2/3$, which corresponds to CC mean length equal 4.

4.1 P2Priv in P2P Scenario

The adversary tries to find out who initiates a content distribution in respect to some content of his/her interest. A linkage between the cloning token's sender—*Alice* and the cloning token is hidden by means of the Mix-net layer. Mix-net is recognized as one of the strongest anonymization method. However, it requires adequate user traffic characteristics to achieve its best results. In particular, traffic volume is crucial for its efficiency. DÃaz et al. in [3] prove that practical Mix designs achieve results close to perfect indistinguishability for high traffic arrivals. For each P2Priv transaction, the P2Priv protocol generates short but numerous messages sent by random nodes. This can allow Mix-net to assure high anonymity without unnecessary delays. The anonymity analysis of Mix-net is independent of the current exposition and in our model we assume perfect performance of Mix-net layer. We assume that the adversary does not get any information about *Alice* during the establishment of CC. However, the adversary having the fraction ρ of malicious nodes can gain awareness that a particular content is about to be distributed. If the cloning token describing this content is passed via one of the malicious nodes, the adversary can disturb CC establishing in a way that allows him to get more information in a subsequent phase of content transportation. The adversary, who looks for the initiator of particular content's distribution, can try to narrow the circle of suspects. While possessing fraction ρ of colluding nodes among N, he/she can break the cloning cascade using the first malicious node which intercepts the cloning token. Let us evaluate the average length of a cloning cascade before the token interception. A set of these nodes will be referred to as CC_{break}. The probability that the adversary manages to break CC immediately after sending out the token by *Alice* equals $\Pr(|CC_{break}| = 1) = \rho$ (we assume that colluding nodes are uniformly distributed among N). The length of random-walk is one more hop longer with probability $\Pr(|CC_{break}| = 2) = (1-\rho)(p_f\,\rho + (1-p_f))$. Then

$$\Pr(|CC_{break}| = n) = (1-\rho)^{n-1}(p_f^{n-1}\,\rho + (1-p_f)p_f^{n-2}). \tag{3}$$

As a result of this action the adversary concludes that the set of parallel connections associated with the interesting content exchange will contain an average of $|CC_{break}|$ links, where

$$|CC_{break}| = \rho + \sum_{i=2}^{|CC|} i(1-\rho)^{i-1}(p_f^{i-1}\,\rho + (1-p_f)p_f^{i-2}) =$$
$$[\,(1+|CC|)p_f{}^{|CC|}(\rho-1)^{|CC|} -$$

$$|CC|\, {p_f}^{|CC|+1}(\rho-1)^{|CC|+1} - \\ p_f(\rho-2) + {p_f}^2(\rho-1)\,] \\ p_f(1+p_f(\rho-1))^{-1}, |CC_{break}| \leq |N|(1\text{-}\rho). \tag{4}$$

After establishing of CC, P2Priv protocol starts direct, plain-text connections from CC members to destination nodes with shared content, each independently delayed with a random interval of time. In the analyzed P2P Scenario the content is shared between equivalent peers N. Then, having in mind the considered threat model, we assume that malicious nodes are also uniformly distributed among destination nodes. The adversary managed to reduce the cloning cascade from mean length equal $|CC|$ to $|CC_{break}|$ (4). Next, he/she will eavesdrop on each content connection established to $\rho\,|N|$ colluded nodes in order to detect transmission of the content of his/her interest and in consequence in search of connected CC_{break} members. It immediately follows that the adversary is able to identify $CC_{eavesdrop}$ peers connected towards the particular content transportation, an average of

$$|CC_{eavesdrop}| = \rho\,|CC_{break}|\,. \tag{5}$$

Until he/she can be certain that $CC_{eavesdrop}$ includes all of the CC or CC_{break} members, he/she cannot determine that $CC_{eavesdrop}$ set includes *Alice*. Let us estimate the uncertainty of the adversary in linking of the found traces to the real initiator. Each $CC_{eavesdrop_k}$, $k = \{1, \ldots, |CC_{eavesdrop}|\}$ node can be *Alice,* with equal probability

$$p_{a1} = \Pr(CC_{eavesdrop_k} = Alice) = |CC_{break}|^{-1}\,. \tag{6}$$

Alice also can be outside eavesdropped nodes and can be each other honest node of the network in the number of $|N| - \rho\,|N| - |CC_{eavesdrop}|$. The attack conducted by the adversary allows him/her to assign probability that one of this nodes is *Alice*, to each equal

$$p_{a2} = \frac{1 - p_{a1}\,|CC_{eavesdrop}|}{|N| - \rho\,|N| - |CC_{eavesdrop}|} = \frac{1 - |CC_{break}|^{-1}\,|CC_{eavesdrop}|}{|N| - \rho\,|N| - |CC_{eavesdrop}|}. \tag{7}$$

Then, we can stress the entropy of P2Priv in P2P Scenario as the following sum of two components corresponding to the set of nodes managed to have been eavesdropped by the adversary and the rest of nodes, respectively

$$\mathcal{H}_{\mathrm{p2priv/p2p}} = -\sum_{k=1}^{|CC_{eavesdrop}|} p_{a1}\log_2(p_{a1}) - \\ \sum_{l=1}^{|N|-\rho\,|N|-|CC_{eavesdrop}|} p_{a2}\log_2(p_{a2}). \tag{8}$$

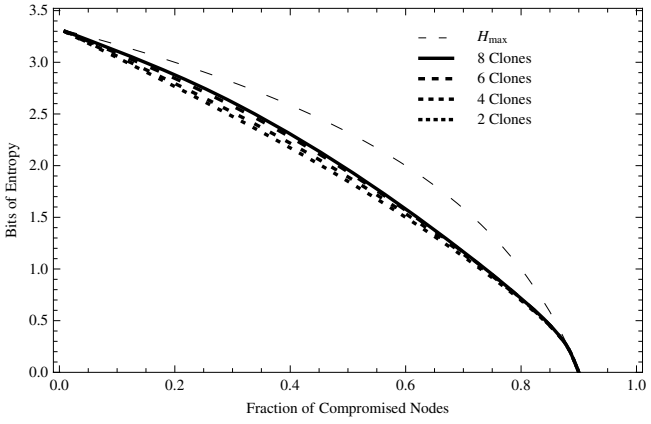


Fig. 2. Entropy for P2Priv in P2P Scenario as compared to maximum entropy, $|N| = 10$

Finally, after substitution and simplification we will then get

$$\mathcal{H}_{\mathrm{p2priv}/\mathrm{p2p}} = \frac{|CC_{eavesdrop}|}{|CC_{break}|}\log_2(|CC_{break}|) - (1 - \frac{|CC_{eavesdrop}|}{|CC_{break}|})\log_2(\frac{1 - |CC_{break}|^{-1}\,|CC_{eavesdrop}|}{|N| - \rho\ |N| - |CC_{eavesdrop}|}). \quad (9)$$

Figure 2 shows entropy of P2Priv architecture as applied to distributed P2P file-sharing service. The entropy of small network ($|N| = 10$) is plotted in the full spectrum of colluding nodes. The presented results were obtained for P2Priv in the following configurations $p_f = \{1/2, 2/3, 4/5, 6/7\}$, which corresponds to mean cloning cascade lengths equal: 2, 4, 6, and 8, respectively. We can observe that, even with a low number of users, P2Priv achieves results close to the maximum in this distributed service scenario and it is robust against a high fraction of compromised nodes.

Figure 3 investigates the robustness of P2Priv in a large scale network comprised by $|N| = 10^3$ nodes. We observe high entropy for a low-to-medium fraction of compromised nodes.

4.2 P2Priv in Client-Server Scenario

Let us apply the P2Priv architecture to the opposite, centralized service scenario. In this case all user nodes N connect to a single server to receive a particular content, which is an *item of interest* of the adversary. We take into account that this server is an easy target for observation and then assume that all its up-link and down-link traffic is being eavesdropped on by the adversary.

Single Request to Server. Similarly to previously analyzed attack scenario, we assume an active adversary who is able to break cloning cascade CC using fraction ρ of malicious nodes scattered in N. Summing it up, we can stress that

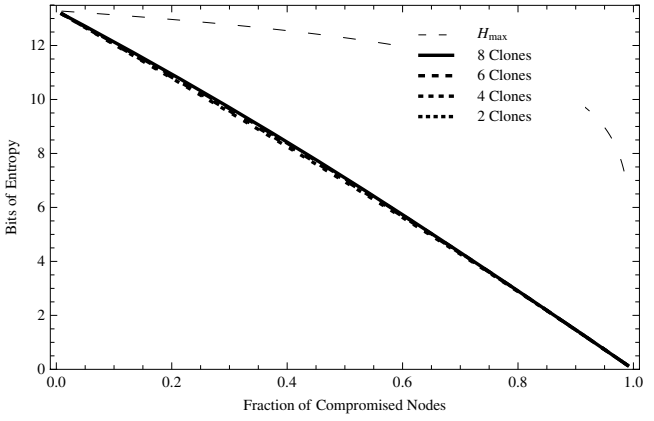


Fig. 3. Entropy for P2Priv in P2P Scenario as compared to maximum entropy, $|N| = 10^3$

in this case, after an initiation of a content connection to the server by *Alice*, all associated connections will be established with the server and eavesdropped on by the adversary. Then, the number of all suspected nodes can be limited by the adversary to $|CC_{break}|$ (4). The adversary knows that one of these nodes belongs to *Alice*, each of them with the probability $p_{a1} = |CC_{break}|^{-1}$ (6). Then, entropy of P2Priv in Client-Server Scenarios is described as follows

$$\mathcal{H}_{\mathrm{p2priv/cs}} = -\sum_{k=1}^{|CC_{break}|} p_{a1} \log_2(p_{a1}) = \log_2(|CC_{break}|). \tag{10}$$

Figure 4 and Figure 5 show entropy of P2Priv architecture applied for Client-Server services. To allow an easier comparison between results obtained for small and large scale networks, the range of entropy plotted in Figure 5 is the same as in Figure 4. Then, it does not include plot for maximum entropy $\mathcal{H}_{\max}$ which certainly is the same as presented in Figure 3. As we expected, the results obtained in the Client-Server Scenario are much worse than those obtained in P2P environment for which P2Priv was intended. Entropy of P2Priv architecture in centralized network is independent of the network scale for a low-to-medium compromised network and takes value of about 1.5 bits for 10%-20% fractions of malicious nodes.

Long-Term Observation. The previous analysis shows entropy of P2Priv in centralized service scenario and the evaluated information leaks correspond to a single connection to the server. However, considering client-server services, sequential requests to a particular server are common. Note that P2Priv does not assure long-term availability of individual nodes, so the CC is deemed to change over time (with the exception of *Alice*). P2Priv was not designed for

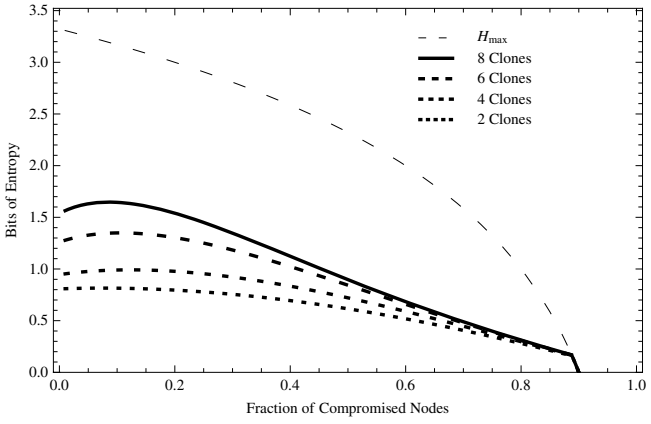


Fig. 4. Entropy for P2Priv in Client-Server Scenario as compared to maximum entropy, $|N| = 10$

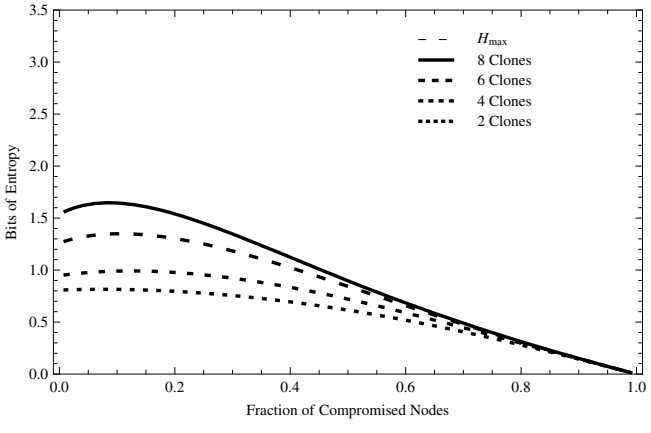


Fig. 5. Entropy for P2Priv in Client-Server Scenario, $|N| = 10^3$

services characterized by a sequential communication to a single network node and it cannot be applied for these purposes in its current form.

4.3 Summary of the Results

P2Priv offers high level of anonymity in P2P environment. However, we have found that the parallel transport architecture of P2Priv does not assure a satisfactory anonymity level for centralized services. Still, the concept of transport parallelization and the moving of time-consuming anonymization techniques to signalization layer seem to be very attractive in the terms of anonymous traffic latency. In the rest of this paper we will discuss the possibilities of using

the parallelism-based approach to network anonymization not limited to P2P content distribution.

5 Network Privacy Preserving Parallel Topology

Considering general purpose anonymous networks, we can distinguish four basic types of network nodes: (i) client/user nodes, (ii) middle-man nodes or proxy servers, (iii) the so called exit nodes, and (iv) service nodes/servers. Client nodes send requests through middle-man nodes to gain an anonymous access to services provided by service nodes (iv). On the other hand, exit nodes (iii) are middle-man nodes which are permitted by their policy to be boundary nodes of forwarding cascades—these nodes connect directly to service nodes (iv) on behalf of users (i). In various networks these sets are merged in different combinations. In particular, all nodes of pure P2P networks can act as client nodes (i), middle-man nodes (ii), exit nodes (iii), and server nodes (iv) as necessary. Such division of roles corresponds to P2Priv. However, when it comes to a generic traffic anonymization, we should separate user nodes from exit nodes as not every user wishes to commit to sharing his/her node as an exit node. Certainly, having in mind client-server network applications, we should also distinguish service nodes (iv).

To provide general-purpose sender anonymization we propose an anonymous network architecture which joins P2Priv parallel transportation of content (collective is comprised by parallel links, similary to P2Priv) with proxy functions (each link is terminated by exit node). The new architecture is derived from P2Priv P2P network and is dedicated to general-purpose anonymous networks, though it will be referred to as NetPriv (network privacy preserving parallel topology).

Let us group nodes in the anonymous network as follows: N is the set of user nodes, potential initiators of communication and E represents exit-nodes. We joined (i) and (ii) in our network model as the anonymity of the proposed solution is basically based on the difficulty of differentiation between real senders and other nodes which simultaneously act the way the senders act. NetPriv is a hybrid solution which largely reflects P2Priv topology of parallel links between N nodes. However, each of these links is additionally terminated by a link to a mixing exit node (E). Figure 6 illustrates the model of the proposed network. In addition, it compares it to classical anonymous networks.

To allow an anonymous communication in the described architecture we propose the following sub-solutions: (i) persistent CC path selection by sender, (ii) time synchronization of requests sending by CC members.

Persistent CC Path Selection. The discussion included in Section 4.2 shows that the parallel architecture of P2Priv, considered in the scope of services characterized by series of user requests to the same destination node or server, is not robust against long-term observations. The reason is that path selection based on a random-walk does not ensure persistent CC paths, so the CC is deemed to

change over time, with the exception of the true sender. The anonymity of CC communications was from the beginning assured by the Mix-net layer. Having this in mind, we propose a replacement of random-walk-based path selection for CC with free-route routing dedicated for Mix-network [7,8]. In this way a sender selects one CC for a whole session of an anonymous communication.

Requests Time Synchronization. The second issue that requires revision is answering the question how exactly CC members randomly delay their connections to destination node/nodes. To assure that the adversary can not distinguish *Alice* from other CC members by an analysis of connection times of individual clones (clone that most frequently connects first can be distinguish as the initiator), we propose inclusion of time field in cloning token which specify starting time for connections and allows synchronization of connection times. The time interval, indicated by *Alice*, should be longer than time required to send token via CC. Then, each CC member adds additional, randomly generated, delay to time indicated in received token. Next, on time calculated in such a way, he/she sends request to an exit node. Accordingly, cloning token will consist of the following fields:

$$\{dest\_addr, reqest\_time, request\}$$

and the request originated to a mixing exit node consists of:

$$\{src\_addr, dest\_addr, request\}_{PK_e}.$$

An exit node sends request to a destination node or nodes, based on information included in received request. Certainly, a source address in his request points at him.

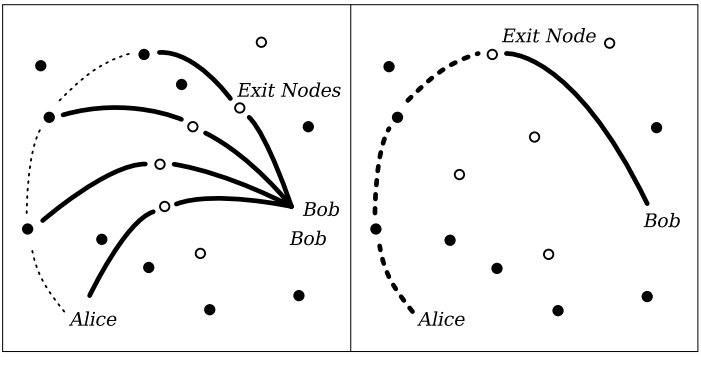


Fig. 6. NetPriv architecture (*left*) as compared to general-purpose classical anonymous network (*right*)

6 Anonymity Analysis for NetPriv

Let us analyze anonymity of the new architecture. Previously, we have shown that the parallel transport architecture assures high entropy for distributed services of file-sharing (8) without improvements proposed in NetPriv. The vital question is how the new architecture of NetPriv deals with Client-Server Service Scenarios (compare to Section 4.2). As the exit nodes and the user nodes have been disjointed, NetPriv anonymity model can allow an analysis of a different fraction of compromised nodes in each of these sets. One can imagine that different service scenarios provoke and permit for different collaboration possibilities, especially as it comes to the set of exit nodes which can be, e.g., dedicated servers of a particular anonymous service, public proxies, or nodes voluntarily provided by network users. As previously assumed ρ represents fraction of malicious user nodes. Let ρ_e represent fraction of malicious nodes among exit nodes. Then

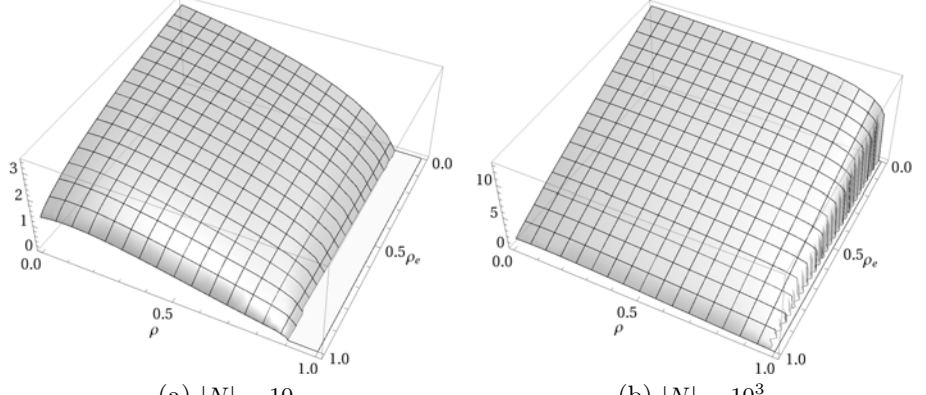
$$|CC_{eavesdrop}| = \rho_e \, |CC_{break}| \, . \tag{11}$$

Similarly to the previous model, the adversary can assign probability p_{a1} to each node of this set (6). *Alice* can also be outside of eavesdropped nodes. We assume that destination nodes/servers are compromised and ρ_e of exit nodes is compromised. Then each honest node of N nodes can be *Alice* with probability

$$p_{a3} = (1 - \frac{|CC_{eavesdrop}|}{|CC_{break}|}) \frac{1}{N - \rho N - |CC_{eavesdrop}|}. \tag{12}$$

Finally, the anonymity of the NetPriv architecture in the Client-Server Scenario is described by entropy

$$\begin{aligned} \mathcal{H}_{NetPriv} = &- |CC_{eavesdrop}| \, p_{a1} \log_2(p_{a1}) - \\ &(N - \rho N - |CC_{eavesdrop}|) p_{a3} \log_2(p_{a3}), \end{aligned} \tag{13}$$



(a) $|N| = 10$ (b) $|N| = 10^3$

Fig. 7. Entropy for NetPriv architect ure, $|CC| = 4$

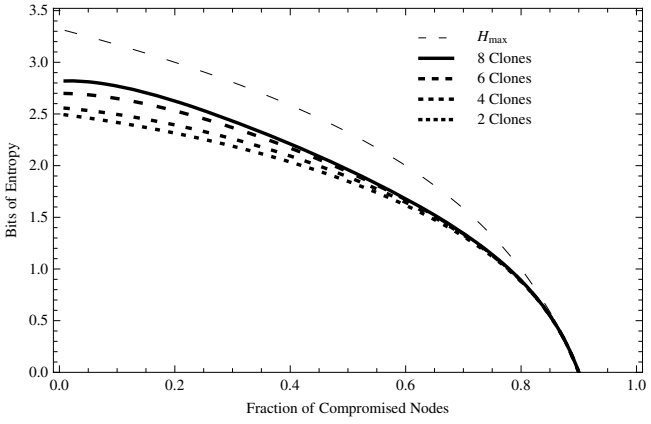


Fig. 8. Entropy for NetPriv in Client-Server Scenario as compared to maximum entropy, $|N| = 10$

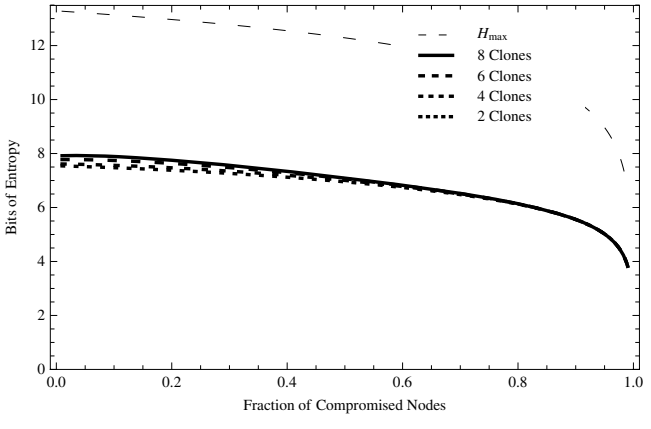


Fig. 9. Entropy for NetPriv in Client-Server Scenario as compared to maximum entropy, $|N| = 10^3$

$$\mathcal{H}_{NetPriv} = \frac{|CC_{eavesdrop}|}{|CC_{break}|} \log_2(|CC_{break}|) - (1 - \frac{|CC_{eavesdrop}|}{|CC_{break}|}) \log_2(\frac{|CC_{break}| - |CC_{eavesdrop}|}{|CC_{break}|\,(N - \rho N - |CC_{eavesdrop}|)}). \quad (14)$$

Figure 7 shows entropy of NetPriv in the full spectrum of compromised user nodes (ρ) and compromised exit nodes (ρ_e) for client-server services.

We can observe that the new architecture assures high entropy both for small as well as large scale networks. Exit nodes can be particularly vulnerable to being compromised. Figures 8, 9 show entropy for NetPriv for constant value of

$\rho_e = 1/2$. The results show that even for this high fraction of collaborating exit nodes NetPriv assure high level of anonymity.

7 Conclusions and Future Work

Today's topologies of anonymous networks shape anonymous communications into hard to trace paths hiding their origin or destination. Transportation of anonymous content via pervasive paths composed of several hops is in favour of anonymity. Still, it imposes a significant traffic bottleneck. The phenomenon of the Internet virtualization and practical possibilities that allow us today to deploy heterogeneous overlay networks of the world-wide range encourage consideration of new network topologies adapted to specific network services. In this paper we have proposed a framework of a novel architecture for anonymous networks—the network privacy preserving parallel topology (NetPriv). NetPriv hides initiators of communications by parallelization of anonymous links. The new approach is based on the premise of the anonymous P2P network called P2Priv [5]. Contrary to P2Priv, the new architecture can be suited to the anonymization of general-purpose network communications. The new solution moves time-consuming anonymization techniques into a signalization layer and combines the primary transport parallelization principle of P2Priv with proxy functions. Additionally, a persistent selection of signalization paths and requests time synchronization mechanisms have been proposed. We applied the information theoretic entropy measurement model to evaluate anonymity of both architectures. We analyzed anonymity of the previous (P2Priv) and the new (NetPriv) architectures with a particular emphasis on centralized, client-server service scenarios. P2Priv can assure high degree of anonymity in the limited scope of network services. P2Priv is dedicated to a large size content distribution and requires a distributed service overlay network to assure high degree of anonymity. These requirement are not met in all network services. We have found that anonymity of P2Priv decreases when destination points of communication are not distributed. Contrary to the previous solution, we have found that the new architecture can be applied to anonymization of various network communications, including client-server services (e.g., anonymous Web access). For a realistic scope of compromised network nodes, NetPriv anonymity is close to maximum. From the user's point of view, the new solution offers a high level of anonymity within only a single proxy node.

The new parallelism-based approach presented in this paper gives the framework for parallel anonymous topologies. This discussion encourages further work. The first vital issue, that needs to be addressed, is the design of follow-up extensions for a bi-directional anonymous communication which includes a receiver anonymity. Secondly, an interesting area of NetPriv analysis is its traffic performance evaluation. Certainly, NetPriv, which allows anonymous transportation via a single proxy, can be considered as the solution able to significantly improve the speed of an anonymous communication (the traffic analysis of P2Priv demonstrates substantial decreases of the content transportation time as compared to

traditional networks [5,16,19]). However, a detailed and practical analysis can be helpful in estimation of NetPriv's working point and its exact impact on network traffic. Finally, other parallelism-based anonymous networks can be considered in a future work. The presented approach within its evaluation can be treated as an impulse for development of further, more complex parallel topologies for anonymous networks.

References

1. Dingledine, R., Mathewson, N.: Anonymity loves company: Usability and the network effect. In: Anderson, R. (ed.) Proceedings of the Fifth Workshop on the Economics of Information Security (WEIS 2006), Cambridge, UK (June 2006)
2. Danezis, G., Wittneben, B.: The economics of mass surveillance and the questionable value of anonymous communications. In: Anderson, R. (ed.) Proceedings of the Fifth Workshop on the Economics of Information Security (WEIS 2006), Cambridge, UK (June 2006)
3. Diaz, C., Sassaman, L., Dewitte, E.: Comparison between two practical mix designs. In: Samarati, P., Ryan, P.Y.A., Gollmann, D., Molva, R. (eds.) ESORICS 2004. LNCS, vol. 3193, pp. 141–159. Springer, Heidelberg (2004)
4. Goldschlag, D.M., Reed, M.G., Syverson, P.F.: Hiding Routing Information. In: Anderson, R. (ed.) IH 1996. LNCS, vol. 1174, pp. 137–150. Springer, Heidelberg (1996)
5. Margasinski, I., Pioro, M.: A concept of an anonymous direct p2p distribution overlay system. In: Proceedings of IEEE 22nd International Conference on Advanced Information Networking and Applications (AINA), pp. 590–597. IEEE Computer Society Press, Los Alamitos (2008)
6. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 4(2) (February 1981)
7. Berthold, O., Pfitzmann, A., Standtke, R.: The disadvantages of free MIX routes and how to overcome them. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 30–45. Springer, Heidelberg (2001)
8. Dingledine, R., Shmatikov, V., Syverson, P.F.: Synchronous batching: From cascades to free routes. In: Martin, D., Serjantov, A. (eds.) PET 2004. LNCS, vol. 3424, pp. 186–206. Springer, Heidelberg (2004)
9. Danezis, G.: Mix-networks with restricted routes. In: Dingledine, R. (ed.) PET 2003. LNCS, vol. 2760, pp. 1–17. Springer, Heidelberg (2003)
10. Böhme, R., Danezis, G., Diaz, C., Köpsell, S., Pfitzmann, A.: Mix cascades vs. peer-to-peer: Is one concept superior? In: Privacy Enhancing Technologies (PET 2004) (2004)
11. Cottrell, L.: Mixmaster and remailer attacks (1994), http://obscura.obscura.com/loki/remailer/remailer-essay.html
12. Gülcü, C., Tsudik, G.: Mixing E-mail with Babel. In: Proceedings of the Network and Distributed Security Symposium - NDSS 1996, pp. 2–16. IEEE, Los Alamitos (1996)
13. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium, San Diego, CA, USA, pp. 303–320 (August 2004)
14. Reiter, M., Rubin, A.: Crowds: Anonymity for web transactions. ACM Transactions on Information and System Security 1(1) (June 1998)

15. Camenisch, J., Lysyanskaya, A.: A formal treatment of onion routing. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 169–187. Springer, Heidelberg (2005)
16. Margasinski, I., Pioro, M.: Low-latency parallel transport in anonymous peer-to-peer overlays. In: Akar, N., Pioro, M., Skianis, C. (eds.) IPOM 2008. LNCS, vol. 5275, pp. 127–141. Springer, Heidelberg (2008)
17. Díaz, C., Seys, S., Claessens, J., Preneel, B.: Towards measuring anonymity. In: Dingledine, R., Syverson, P.F. (eds.) PET 2002. LNCS, vol. 2482, pp. 54–68. Springer, Heidelberg (2002)
18. Serjantov, A., Danezis, G.: Towards an information theoretic metric for anonymity. In: Dingledine, R., Syverson, P.F. (eds.) PET 2002. LNCS, vol. 2482, pp. 41–53. Springer, Heidelberg (2003)
19. Margasinski, I.: Anonymous Transport in Peer-to-Peer Overlay Networks. PhD thesis, Warsaw University of Technology (June 2008)

Security Usability of Petname Systems

Md. Sadek Ferdous[1,2,3], Audun Jøsang[2,3],
Kuldeep Singh[1,2,5], and Ravishankar Borgaonkar[2,5,6]

[1]NTNU, [2]UNIK, [3]University of Oslo, [4]University of Tartu, [5]TKK, [6]KTH
{sadek,josang,kuldeep,ravishankar}@unik.no

Abstract. To have certainty about identities is crucial for secure communication in digital environments. The number of digital identities that people and organizations need to manage is rapidly increasing, and proper management of these identities is essential for maintaining security in online markets and communities. Traditional Identity Management Systems are designed to facilitate the management of identities from the perspective of the service provider, but provide little support on the user side. The difficulty of managing identities on the user side causes vulnerabilities that open up for serious attacks such as identity theft and Phishing. Petname Systems have been proposed to provide more user friendly and secure identity management on the user side. This paper provides an analysis of the Petname Model by describing its history and background, properties, application domains and usability issues with emphasis on Security Usability. By covering a broad set of aspects, this paper is intended to provide a comprehensive reference for the Petname System.

1 Introduction

The purpose of digital communication protocols is to exchange information as efficiently and reliably as possible. Originally, these protocols were designed without authentication because the identities of communicating parties could be assumed, and did not have to be formally verified. Authentication was subsequently added for verifying the correctness of claimed and assumed identities. Authentication requires prior registration of identities, and is based on a set of security mechanisms combined with a credential or security token. As authentication became necessary for accessing many online services, more and more identities and credentials were issued, and their management became problematic, both for service providers and for users. Identity Management (IdM, in short) was introduced by the industry to facilitate server-side management of user identities. Initially, client-side management of user identities was not considered to be an issue. However, many people currently feel overloaded with identities and passwords that security policies require them to memorize. The growing number of identities that users need to handle and the inability of users to comply with credentials management policies now makes client side IdM a critical issue. It is also important to consider that SP (Service Provider) identities also need to

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 44–59, 2009.

be managed, and this aspect of IdM has received very little attention. Petname Systems, which we will discuss here are precisely focused on client-side management of SP identities. We would like to point out here that we will use the term Petname Model to denote the abstract properties of Petname Systems. An implementation of the Petname Model is then a Petname System.

To understand the Petname Model it is essential to understand why Petname Systems were proposed in the first place. The Petname Model was formally described by Marc Stiegler in his 2005 paper [1]. The potential of the Petname Model, however, was discovered by several people in several successive steps. Elements of the idea behind Petname Systems are scattered among several papers and web articles, and the combined efforts of these authors have shaped the formulation of the Petname Model. Section 2 aims to summarize the existing literature.

Section 3 defines the Petname Model by outlining its different components and establishing the connections among them. A Petname System can have several properties and its potential applications can span over several disciplines of computing and networking. A long list of properties as well as several application scenarios were listed in [1]. Section 4 formalizes the properties in a more systematic way by dividing them into two broad categories: 1) Functional properties and 2) Security Usability properties, and also by adding new usability requirements. Security Usability of Petname Systems will be analyzed in Sect. 5. In Sect. 6, different applications of the Petname Model are explained. The current paper introduces some new application scenarios other than those discussed in [1]. Section 7 analyzes the usability issues of two applications that utilize the Petname Model. Section 8 provides some hints on potential future work on Petname Systems and concluding remarks are provided in Sect. 9.

2 Background and Rationales of Petname Systems

At first, it is important to note the relation between Entity, Identity and Identifier. In the scope of this paper, a person, an organization or a machine (computer) operated by any person or organization will be denoted as entity. Identity is the *fundamental property of any entity that declares the uniqueness or sameness of itself and makes it distinctive from other entities in a certain context* [2]. In general, an entity can have multiple identities, but an identity cannot be associated with more than one entity. Each identity can consist of multiple attributes that are also known as identifiers when used for identification purpose [3]. Here, the same attribute can be associated with multiple identities.

Three desirable properties of an identifier were defined by Zooko Wilcox-O'Hearn in his influential web article published in 2001. According to Wilcox-O'Hearn an identifier should ideally be Global[1], Unique[2] and Memorable[3] [4,5]. To be memorable, an identifier has to pass the so-called "moving bus test" [6].

[1] Called "Decentralized" in [4].
[2] Called "Secure" in [4].
[3] Called "Human-Meaningful" in [4].

That is, if one can correctly remember a name written on a moving bus for a definite amount of time, that name can be considered memorable. An identifier will be unique if it is collision-free within the domain [1] and has the property that it is strongly resistant against forgery. Wilcox-O'Hearn also claimed with supporting evidence that no identifier could have all the three desirable properties simultaneously, and suggested to choose any two of them according to different scenarios.

A triangle where the three properties are placed in the three corners is commonly known as Zooko's triangle, and represents the basic foundation for the Petname Model. Zooko's triangle is illustrated in Fig. 1.

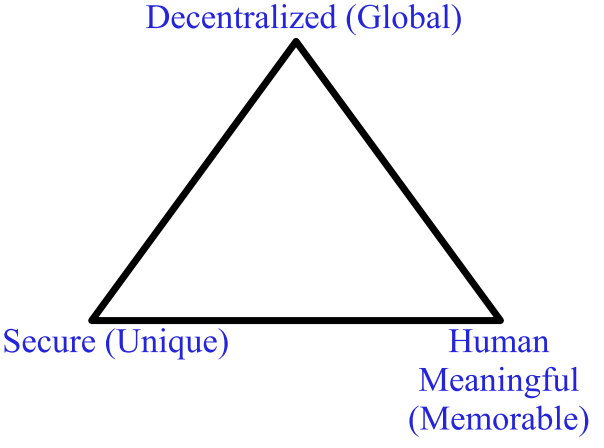


Fig. 1. Zooko's triangle

The idea of placing the three properties at the three corners of a triangle can be explained as follows. In a triangle the three corners are never connected by a single line, only pairs of corners are connected. Placing those three properties in the three corners of the triangle provides a visual analogy to the fact that an identifier can only achieve two of the desirable properties at any one time.

In 2000, Jonathan S. Shapiro, being inspired by the idea of Marc Miller *et al.* while at Electric Communities, described in a web article his scheme of adopting a system which utilized three types of naming conventions: *Petname*, *True Name* and *Nickname* [7]. He adopted this idea for a configuration management system. A True Name is synonymous to a globally unique identifier, the Nickname is a globally memorable name of an entity assigned by its creator, and the Petname is a memorable and locally unique user-assigned name for that entity.

A few months later, Mark Miller published another article [6] in which he, for the first time, documented the structure of the Petname Model with three components: Petname, Key and Nickname. These three components are essentially equivalent to Shapiro's Petname, True Name and Nickname respectively. Miller suggested to use the term Key instead of True Name, and pointed out that the Petname Model satisfies all the three desirable properties of Zooko's triangle.

Tyler Close suggested to adopt the term Pointer instead of Key [8] and we will also use the term Pointer in this paper.

In 2003, Tyler Close of Waterken Inc. pointed out the possibility of using Petname Systems for better trust management [9]. Waterken Inc. developed the Petname Toolbar for the Firefox web browser. The main motif was to show the potential application of Petname Systems to counter phishing attacks. According to Tyler Close, humans are not capable enough to manage the transition of trust from one entity to another in digital communication and this leads to identity-theft as a result of phishing attacks and suggested to use Petname Systems to enable manual trust evaluation by the user while the transition takes place.

In 2005, Marc Stiegler extended the Petname Model based on Mark Miller's suggestion and also explained the detailed interaction among the components of the Petname Model [1]. He also formalized the properties and requirements for the Petname Model and gave examples of some applications of Petname Systems.

3 The Petname Model

3.1 Rationale

As mentioned in the previous section, Zooko's triangle visualizes the hypothesis that no identifier can be Global, Memorable and Unique at the same time, but can only have two properties. Three unique pairs can be created using these three properties: 1) Global-Memorable, 2) Memorable-Unique and 3) Global-Unique. Even if no identifier can have all the three properties, a *naming system* can be designed to achieve all the three properties of Zooko's triangle. The Petname Model represents one such naming system.

3.2 Components

The Petname Model uses three different types of names that in our terminology are called: Pointer, Nickname and Petname. These three names actually represent the three sides of Zooko's triangle and hence are synonymous to the three pairs discussed above. Detailed explanation for each of them are given below.

Pointer. The Pointer was defined as "True Name" in Shapiro's interpretation and as "Key"in Miller's interpretation. A Pointer implies a globally unique and securely collision free identifier which can uniquely identify an entity. It interconnects the *Global* and *Unique* corners of Zooko's triangle. The security of the Petname Model largely depends on the difficulty to forge a Pointer. A public/private key pair and a fully qualified pathname of a file in an Internet file server are good examples of Pointers. They are globally unique and difficult to forge. However, a Pointer (e.g. a public key, IP address, etc.) may not be memorable to human.

Nickname. The Nickname inter-connects the *Global* and *Memorable* corners of Zooko's triangle. It is an optional non-unique name created by the owner of the

Pointer. The purpose of the Nickname is to aid in identifying the entity easily. The title of a web page that is displayed in the title bar of the browser is an example of a Nickname. Users may remember that webpage by the title, but another website may have the same title and can create a collision on the user's mind. Thus a Nickname is not necessarily unique.

Petname. The Petname is a name created by the user to refer to a specific Pointer of an entity. Within the domain of a single user a bidirectional one-to-one mapping exists between Petnames and Pointers. A Petname connects the *Memorable* and *Unique* corners of the triangle. Petnames only have a local scope and may only be relevant for local jurisdiction. The same Petname can be used by different users to refer to either the same Pointer or to different Pointers. The security of a Petname System also depends on the privacy of Petnames and the difficulty to mimic a Petname. Here it is interesting to note that a Petname does not necessarily mean a text-based name. In addition to text, it can also be image and sound or any combination of them in different ways.

3.3 Relationship among the Components

There is a bidirectional one-to-one mapping between Pointers and Petnames within the domain of each user. A Nickname has a one-to-many relationship to the set of Pointers. A Pointer is assumed to map to a single Nickname, but can map to several Alleged Names in the global domain. An *Alleged Name*, like the Nickname, is the introductory/referred name for an entity given by a third party. The distinction between a Nickname and an Alleged Name is that the Nickname is created by the owner of the entity and the Pointer whereas the Alleged Name is provided by a third party. The relationship between Petnames and Nicknames can be confusing sometimes. In some situations, a Nickname can be used as a Petname or in other situations a Petname can be derived from the Nickname. A single Nickname can always be uniquely resolved from the Petname, but the Nickname is not necessarily unique for the Petname. For that reason, a Petname can not be uniquely resolved from a Nickname. Figure 2 illustrates this relationship. As seen from the figure, the Petname Model is actually a naming convention built on top of Zooko's triangle.

4 Properties of Petname Systems

The properties of a Petname System can be divided into two broad categories: Functional properties and Security Usability properties.

4.1 Functional Properties

Functional properties are those basic properties that are mandatory for a Petname System. The functional properties are [1]:

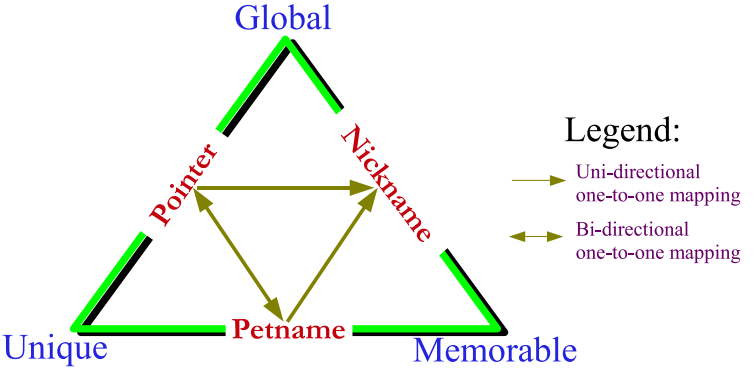


Fig. 2. The Petname Model

F1. A Petname System must consist of at least a Pointer and a Petname.
F2. Nickname is optional.
F3. Pointers must be strongly resistant against forgery so that the Pointer can not be used to identify a false entity.
F4. For every user there must be a bi-directional one-to-one mapping between the Pointer and the Petname of each entity.

4.2 Security Usability Properties

Security usability will ensure the reliability of using the system and enables the user to draw conclusion on the actual security of the system. These properties will ensure that the Petname System is not affected by usability vulnerabilities. Usability properties can again be categorized in two types [10]:

1. A security action is when users are required to produce information and security tokens, or to trigger some security relevant mechanisms. Security action enables a user to interact securely with an entity. For example, typing and submitting a password is a security action. Properties related to the security action in the Petname System are [1]:
   SA1. It is the user who must assign the Petname for each Pointer.
   SA2. Users must assign the Petname for the Pointer with explicit action.
   SA3. As the relationship between the user and other entities evolve, the user should be able to edit the previously applied Petname for a Pointer to a new Petname.
   SA4. Suggestion on the Petname based on the Nickname can be provided as an aid for the user to select a Petname for a Pointer. If the Nickname is missing, other criteria could be chosen for the suggestion.
   SA5. If a suggestion is provided and the user wants to accept it as the Petname, then he must do so with explicit action.
   SA6. Petname Systems must make sure that the user-selected, created or suggested Petname is sufficiently distinct from the Nickname so that the user does not confuse them with each other.

SA7. Petname Systems must make sure that the user-selected, created or suggested Petname must be sufficiently different from existing Petnames so that the user does not confuse them. This is needed to reduce the risk of mimicry of the Petname upon which the security of the Petname System largely depends.

SA8. If the user chooses a Petname that may resemble a Nickname or other Petnames, he should be warned explicitly.

SA9. The User should be alerted to apply a Petname for the entity that involves in highly sensitive data transmission.

2. A security conclusion is when users observe and assess security relevant evidence in order to derive the security state of systems. Security conclusions enable the user to conclude on the security state of the system by observing security relevant evidence and assessing this together with assumptions. For example, observing a closed padlock on a browser, and concluding that the communication is protected by TLS is a security conclusion. Properties related to the security conclusion are [1]:

SC1. The Pointer and the corresponding Petname must be displayed at all times through the user interface of the Petname System. This will make the user confident about his interaction and help to draw the security conclusion easily.

SC2. The Petname for a Pointer should be displayed with enough clarity at the user interface so that it can attract the user's attention easily.

SC3. The absence of a Petname for a Pointer should be clearly and visually indicated at the user interface so that the user is surely informed about its absence.

SC4. The visual indication for suggested Petnames and Nicknames should be unambiguous enough so that the user does not confuse them with each other.

SC5. The warning message that will be provided when there is a direct violation of any of the above properties should be clear enough so that the user can understand the problem and take the necessary security action.

5 Evaluation of Security Usability for Petname Systems

The usability of security is crucial for the overall security of the system, but is still a relatively poorly understood element of IT security. Therefore it is important to evaluate the Security Usability of Petname Systems as it is directly related to the security of client-side Identity Management. A set of general Security Usability principles related to Identity Management were proposed in [10]. We will use these principles as a basis to evaluate the Security Usability of the Petname System by analyzing if the Security Usability properties of the Petname System satisfy these principles. The Security Usability principles are described below:

Security Action Usability Principles:

A1. Users must understand which security actions are required of them.
A2. Users must have sufficient knowledge and the ability to take the correct security action.
A3. The mental and physical load of a security action must be tolerable.
A4. The mental and physical load of making repeated security actions for any practical number of instances must be tolerable.

Security Conclusion Usability Principles:

C1. Users must understand the security conclusion that is required for making an informed decision.
C2. The system must provide the user with sufficient information for deriving the security conclusion.
C3. The mental load of deriving the security conclusion must be tolerable.
C4. The mental load of deriving security conclusions for any practical number of instances must be tolerable.

The Security Usability properties of Petname Systems can now be analyzed according to these security principles. When a Petname System satisfies SA1-SA3 and SA6-SA9 of the Security Action properties, it implicitly implies that principles A1 and A2 are also satisfied, because the former properties enable a user to select a unique and unambiguous Petname for a Pointer. This selection of a unique and unambiguous Petname for a Pointer can be thought of as the correct security action as it enables the user to securely identify an entity. Security Action properties SA4-SA8 will act as the aid for the user to select a Petname for a Pointer. We believe that selecting an unambiguous Petname will pose the most significant mental load for the user in the Petname System when repeated for several entities. Such mental load will be reduced significantly if these five properties are satisfied in a Petname System because users do not have to think about the ambiguity of the new Petname with other existing Petnames. Automated suggestion could also be a great aid in such selection. Therefore satisfying these five properties will implicitly lead to the principles A3 and A4 also being satisfied.

To analyze the Security Conclusion properties of the Petname System, we have to first define Security Conclusion in the Identity Management perspective. Security Conclusion in the Identity Management perspective is to correctly identify a specific entity. Displaying the Petname for a Pointer that points to the desired entity at the user interface will enable the user to draw conclusion that this Pointer and in turn the entity the user is interacting with is the intended one. The presence and absence of the Petname will provide the user with enough information to draw the security conclusion easily. So whenever a Petname System satisfies SC1-SC3, it will explicitly satisfy C1 and C2. Different visual techniques should be applied to help the user reduce their mental load in deriving security conclusion. Using different eye-catching colors to indicate the presence or absence of a Petname for a specific Pointer can be an example of

one such visual technique. The security conclusion properties SC2-SC5 should be applied to enable a user to draw conclusion with ease and thus if followed will satisfy principles C3 and C4.

From the above analysis we can conclude that a complete implementation of all the properties of a Petname System will satisfy all the security usability principles.

6 Application Domains

The presence of the Petname Model is so ubiquitous that people may sometimes be unaware of its existence. Here we will highlight the possible domains in which the Petname Model is used, intentionally or unintentionally, or could be used. For each of the applications we will try to determine the suitability of applying the Petname Model [1].

Real World. The principle of the Petname Model is so naturally integrated in the real world that we do not notice its existence. Let us first analyze how people actually recognize each other. This process is very simple and natural to us: through several physical attributes like face, voice, physique or maybe combinations of them. These combinations can be thought of as the Pointer to uniquely identify a single person. That single person introduces himself to us by stating his name XYZ which is actually a Nickname in the Petname Model terminology. From then on we may perceive that man's identity as Mr. XYZ, which actually represents a Petname. Now if another person also introduces himself as XYZ, then our mind does not only assign that name as his Petname because it was already assigned to another person. Here things may evolve in different directions. One possible direction can be that our mind distinguishes between those two persons and changes the Petname for the first person as Mr. XYZ of London and Mr. XYZ of Paris for the second person or whatever seems practical.

Phone/E-mail Contact List. A phone/email contact list is another classic example of a Petname System. The phone number with international format (preceding the number with + or 00 and country code) may represent the Pointer and it is unforgeable and globally unique. We save the number in our contact book by placing a name for it which is nothing but a Petname for that number. Nicknames are absent here. The same analogy applies for email contact lists. Email addresses represent Pointers. A *From-field* in an email header may contain only the email address: xyz@yahoo.com or a given name by the sender with his email address: Mr. XYZ <xyz@yahoo.com>. Here the given name (Mr. XYZ) represents the Nickname. After receiving a mail from a new sender one can save the sender's email address in the email contact list. At that time a Petname is created by inserting a name suitable to identify that person, or by simply keeping the Nickname.

IM Buddy List. In the domain of a particular Instant Messaging Service each entity has a unique Id (email Id for yahoo, hotmail or passport service)

which represents the Pointer for that entity. But sometimes those Ids can have quite close resemblance (logicman and 1ogicman, the second one actually is a 1 not a small L) to each other and thus can be quite confusing for the user to differentiate. A better option is used in the interface of the Instant Messenger where one can put a name for each of the IDs. Such name is actually a Petname. In the user interface all the interactions with the ID is usually done with the Petname and thus making the IM Buddy list a good example of a Petname System. Nicknames are absent here.

DNS & Anti-Phishing Tool. As mentioned earlier, that two domain names can be quite close to each other (typo squatting), intentionally or unintentionally, which can lead to phishing or pharming attacks, Petname Systems can be a useful tool to thwart this type of attack. The domain name itself represents the Pointer. The title in the title bar of the browser for that domain name is the given Nickname. In the user interface (the browser), the user can provide a Petname for each domain name. All the interactions with that domain will be indicated by the Petname in the user interface. Providing a Petname for each domain name will impose a trust relationship to that domain name. Absence of Petname will indicate the absence of a trust relationship.

On the background of the above scenarios, the typical e-commerce transaction scenario can be analyzed. A user frequently shops online and places his trust in PayPal to process his online transaction. Now to safely process his transaction he can define a Petname for PayPal in his browser. Assume that the user visits an e-commerce site that offers an item he wants to buy, but the users does not trust the site to know his credit card details. Luckily the site allows him to pay through PayPal, so he is redirected to `www.paypal.com` when the transaction enters the payment phase. Assuming that he has already defined a Petname for PayPal, his browser should indicate the Petname for it and he feels confident that it really is PayPal, and authorizes the transaction. Assuming that the e-commerce site is fraudulent, and redirects him to `www.paypa1.com` (note that it is "1", not a small "L") to phish him, his browser will not find a corresponding Petname because the domain name does not match. The missing Petname will alert him that the PayPal site is fraudulent, and that he should abort the transaction.

This idea has been utilized in three Firefox extensions to develop a defense mechanism against Phishing Attack. These are: the Petname Tool [11], developed by Tyler Close, TrustBar [12], developed by the TrustBar team at the Dept. of Computer Science in the Bar Ilan University, Israel and Passpet [13], developed at the CYLAB of the Carnegie Mellon University.

IP Address. Not all IP addresses have domain names. If one would like to communicate only utilizing IP address, a Petname Model can be applied locally as a substitute for domain names. IP addresses are hard to remember, and Petnames will make it easy to refer to them. IP addresses will represent the Pointer, and the corresponding Petname will be used at the user interface. All communication from the user's side will be based on Petnames.

Process Handling. Every modern OS runs a number of processes simultaneously. *ps -e* command in Linux or the process tab in the task manager for

Windows shows a long list of processes. Some of the process names are so obscure that it is impossible for the user to understand their functionality. A Petname Model can be applied to improve the situation significantly. When a process runs for the first time it will present a short description of what it will do. Then the user can create an informative Petname for that process. This Petname will be displayed in the memory map, for example in the process tab in task manager or with *ps -e* command. In this case the Pointer does not have to be global. It is simply the unique process name or unique command used to run the process.

7 Evaluation of Security Usability for Petname System Applications

Having formalized the properties of Petname Systems, and having analyzed security usability issues on a general level, the security usability for two existing Petname System applications are analyzed with the Cognitive Walkthrough method. The applications to be analyzed are : 1) Petname Tool and 2) TrustBar. Both toolbars are designed only to work with the Firefox browser, and are aimed at simplifying client-side management of SP identities and at providing a better defense mechanism against Phishing attacks. Though the application domains for the Petname System is much broader, as described in Sect. 6, we have decided to confine our evaluation only to these two in order to focus on managing SP identities at the client side. These two particular applications exactly meet this criterion.

The Cognitive Walkthrough method is a usability evaluation method in which an evaluator or a group of evaluators participate to identify the usability issues of an application by visually inspecting the user interface. Because Petname Systems affect the user interface, Cognitive Walkthrough is a suitable method for evaluating their usability. While performing the Cognitive Walkthrough for each application, we will try to note if the application satisfies the usability properties discussed in Sect. 4. The degree of compliance with the specified security usability properties will give an indication of the level of security usability of each application. For the evaluation we will be using Firefox version 3.0.7 with Nightly Tester Tool, a Firefox add-on, installed.

7.1 Evaluation of the Petname Tool

The Petname Tool is available as a Firefox add-on in [14]. Once installed the toolbar will look like Fig.3. The Petname Tool is very simple, however, one may almost feel that it is too simple. It does not come with any text label; only a text field to enter Petnames. Absence of a text label can confuse unfamiliar users because they might not understand its purpose. The Petname Tool does not work for non-https sites, therefore it will not be possible for a user to assign Petnames to non-https sites. Another thing is worth to note that the Petname Tool uses the hash of the public key of a website as a Pointer. Therefore if the site receives a new certificate and thus a new public key, the Petname Tool will fail to

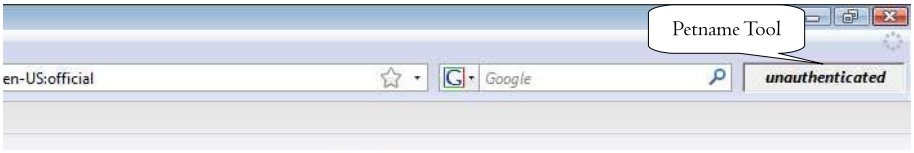


Fig. 3. The Petname Tool in Firefox

map between the already assigned Petname and the Pointer. A possible solution could be to let URL or domain name be the Pointer that will also remove the restriction of applying Petnames for https sites only.

In the following, the Petname Tool will be analyzed for compliance with the Petname System properties. The Petname Tool, obviously, deploys Petnames. The hash of the public key, derived from the certificate, represents the Pointer and is strongly resistant against forgery. Therefore we conclude that the Petname Tool satisfies F1 and F3. But a serious restriction of the Petname Tool is that it allows users to assign exactly the same Petname for different entities, thus violates the principle of bi-directional one-to-one mapping for each entity and therefore also violates F4. It does not deploy Nicknames and therefore does not satisfy the optional property F2.

The Petname Tool enables users to explicitly assign a Petname for each entity, e.g. to select the text field, write down a Petname and hit the Enter key. This satisfies SA1 and SA2. Users can change any Petname any time, thereby satisfying SA3. No suggestion is provided for aiding the user to select a Petname, which is not compliant with SA4 and SA5. Also Nicknames are not used in the Petname Tool, resulting in non-compliance with SA6. Whenever a user selects a Petname that closely resembles or similar to the existing Petnames, the user is alerted with an informative dialog box (Fig.4). The dialog box displays the existing Petnames to which the current Petname has close resemblance. The user can ignore the alert by clicking the *Assign petname* button or he can cancel this current Petname by clicking the *Don't assign petname* button. If the new Petname is similar to the existing one and he clicks Assign Petname button, the same Petname will be displayed for both websites when he visits them later. Therefore, the Petname Tool is compliant with SA8 (showing the two dialog boxes with the warning), but directly violates SA7. The Petname Tool does not show any alert when there is highly sensitive data transmission and therefore indicates the absence of SA9.

The Petname, if already supplied by the user, is displayed on the Petname Tool toolbar, thereby satisfying SC1. Different typefaces, tooltips and colors have been used in the Petname Tool to catch the user attention to indicate the presence or absence of a Petname. In our opinion, white and light green as used by the Petname Tool is less visible than Red, Yellow or Green, as suggested in [15]. In addition, blinking text or different text colors could be used to draw more user attention. Nevertheless, we can conclude that the Petname Tool is compliant with SC2 and SC3. As there is no suggested Petname or Nickname in the Petname Tool, it does not satisfy SC4. The Petname Tool provides warning

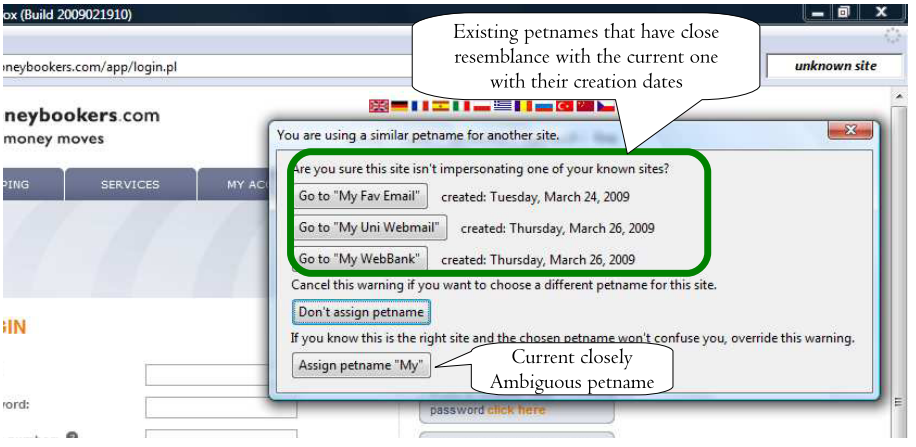


Fig. 4. Dialog box that warns the close ambiguity/similarity among different Petnames

through dialog boxes when there are conflicts with other Petnames or if there is ambiguity between Petnames and thus satisfies SC5. However, it does not provide a warning message when there is a violation for other properties.

Apart from security usability issues, there are some other weaknesses in the Petname Tool. For example, there is no help button that could explain what the user has to do to utilize it properly. It does not provide the standard *About* menu item that could explain the purpose of the Petname Tool.

7.2 Evaluation of the TrustBar

The TrustBar Tool is available as a Firefox add-on in [12]. The current version of TrustBar is not compatible with the latest Firefox version. Therefore the Nightly Tester Tool, another Firefox add-on, was used to resolve the compatibility issues. Once installed the toolbar looks like Fig.5.

TrustBar overcomes some of the shortcomings of the Petname Tool. For example, it works for non-https sites, provides an excellent *Help* feature and also comes with the standard *About* menu item that provides a short description of what it does.

The following simple analysis of TrustBar gives an indication of how it satisfies the properties of the Petname Model. TrustBar utilizes Petnames, and thereby complies with F1. The domain name or URL represents the Pointer and is strongly resistant against forgery. Therefore we conclude that TrustBar satisfies F1 and F3. TrustBar also displays a Nickname in the form of the organization name, if a certificate is available or in the form of the domain name for non-https sites and thus satisfies F2. However, a serious restriction of TrustBar is that it allows users to assign exactly the same Petname for different entities as demonstrated in the next paragraph, thus violates the principle of bi-directional one-to-one mapping for each entity and therefore violates F4.

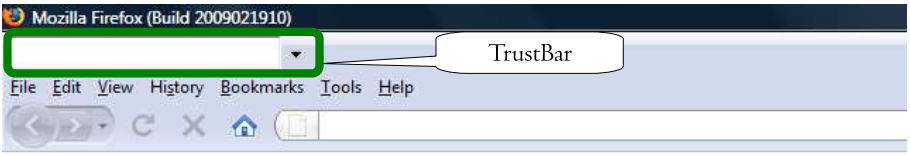


Fig. 5. TrustBar installed in Firefox

TrustBar enables a user to assign a Petname for each entity so he has to act explicitly, e.g. select the text field, write down a Petname and hit the Enter key, to enable the Petname and this satisfies SA1 and SA2. Users can change any Petname any time and thus TrustBar meets the requirement of SA3. A suggestion is provided in the form of a Nickname for aiding the user to select a Petname if a server certificate is available and this satisfies SA4 partially, and the user has to act explicitly, e.g. by hitting the Enter key so the text field turns to light green (an indication for accepting the Petname) to accept the Nickname as the Petname. This satisfies SA5 too. A serious restriction of TrustBar is that it allows users to assign ambiguous Petnames or even equal Petnames for different entities. It does not provide any sort of warning to users about the ambiguity or similarity of the Petnames and thus directly violates SA7 and SA8. TrustBar allows a user to assign a Petname at his will whenever he feels and does not show any alert when there is highly sensitive data transmission and therefore violates SA9.

The Pointer and the related Petname in the TrustBar, if already supplied by the user, are displayed all the time in the browser toolbar, thereby satisfying SC1. Different icons, tooltips and colors have been used in the TrustBar to catch the user attention to indicate the presence or absence of Petnames. In our opinion it would have been better to use more flashy colors like Red, Yellow or Green instead of pale yellow and light green. Blinking text or different text colors could be used to draw more user attention to potential security problems in websites. Nevertheless, we can conclude that TrustBar satisfies SC2 and SC3. White, pale yellow or light green color has been used to differentiate among non-https Nicknames, https Nicknames and Petnames respectively, thereby satisfying SC4. TrustBar does not provide any sort of warning to the user and this indicates the complete absence of SC5.

7.3 Summery

Table 1 clarifies the distinction between the Petname Tool and TrustBar in terms of the properties of Petname Systems. It can be noted that TrustBar satisfies more properties of the Petname model than the Petname Tool, though TrustBar has one major shortcoming: absence of any type of warning message. Both tools suffer from the absence of the crucial property F4. As neither of them satisfy all the main properties of Petname Systems, we can conclude that none of them fully satisfies the security usability principles.

Table 1. Comparison between the Petname Tool and TrustBar

	F				SA									SC				
ToolName	1	2	3	4	1	2	3	4	5	6	7	8	9	1	2	3	4	5
Petname Tool	**Y**	N	**Y**	N	**Y**	**Y**	**Y**	N	N	N	N	**Y**	N	**Y**	**Y**	**Y**	N	**Y**
TrustBar	**Y**	**Y**	**Y**	N	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**	N	N	N	**Y**	**Y**	**Y**	**Y**	N

8 Future Work

No tool is currently available for Petname based identity management that satisfies all the properties of Petname Model and the corresponding security usability principles, as indicated by our analysis in the previous section. Developing a Petname Model based tool that satisfies the security usability principles should be a priority in future research and development.

As the Petname Model is based on Zooko's triangle, any shortcut in the triangle may collapse the relationships among the components of the Petname Model or may create a new dimension of relationship. Bob Wyman in his web blog proposed to update Zooko's triangle into a pyramid by inserting a new attribute called "Persistent" and connecting it to the other corners. The new attribute was proposed to signify the longevity of each name [16]. This proposal to change the shape of Zooko's triangle can be another potential topic for research which could give Petname Systems additional security properties.

Smart phones are becoming increasingly popular and the number of people that access the Internet from their smart phones is growing every day. Investigations on how the Petname Model can be implemented and adapted for the tiny screen of a mobile phone can be a challenging task and thereby another scope for future research.

9 Conclusions

The Petname Model is naturally embedded in human perception to identify different entities. Implementing it in computer networks and system is a natural extension of human cognitive capabilities and represents a great aid for humans in digital environments. This fact has been demonstrated through several applications, experiments and proposals. A large scale adaptation of the Petname Model is therefore timely.

This paper has focused on providing a brief overview of Petname Systems, formally defined the properties of Petname Systems and explained how these properties can satisfy essential security usability principles. We believe that if these properties are followed in developing applications based on the Petname Model, it will improve the overall security by removing security vulnerabilities related to poor usability. The paper has also analyzed two available Petname-based applications and shown that they represent an improvement in usability, but unfortunately do not satisfy all the specified security usability principles.

References

1. Stiegler, M.: Petname systems (August 2005), http://www.skyhunter.com/marcs/petnames/IntroPetNames.html (Last visit on May 20, 2009)
2. Thanh, D.V., rstad, I.J.: The ambiguity of identity. Telektronikk issue on Identity Management 103, 3–10 (2007)
3. Jøsang, A., Pope, S.: User centric identity management. In: Asia Pacific Information Technology Security Conference, AusCERT 2005, Austrailia, pp. 77–89 (2005)
4. Wilcox-O'Hearn, Z.: Names: Decentralized, secure, human-meaningful: Choose two (2005), http://www.zooko.com/distnames.html (Last visit on May 30, 2009)
5. Internet archive wayback machine:snapshot on zooko's writing (2008), http://web.archive.org/web//http://zooko.com/distnames.html
6. Miller, M.: Lambda for humans (2000), http://www.erights.org/elib/capability/pnml.html (Last visit on May 30, 2009)
7. Shapiro, J.S.: Pet names, true names, and nicknames (2000), http://www.eros-os.org/~majordomo/dcms-dev/0036.html (Last visit on May 30, 2009)
8. Close, T.: Naming vs. pointing (2003), http://www.waterken.com/dev/YURL/Analogy/ (Last visit on May 30, 2009)
9. Close, T.: Waterken YURL:trust management for humans (2003), http://www.waterken.com/dev/YURL/Name/ (Last visit on May 30, 2009)
10. Jøsang, A., Al Zomai, M., Suriadi, S.: Usability and privacy in identity management architectures. In: Brankovic, L., Steketee, C. (eds.) Fifth Australasian Information Security Workshop (Privacy Enhancing Technologies) (AISW 2007), Ballarat, Australia. ACS. CRPIT., vol. 68, pp. 143–152 (2007)
11. Close, T.: Petname tool: Enabling web site recognition using the existing SSL infrastructure (2006), http://www.w3.org/2005/Security/usability-ws/papers/02-hp-petname/ (Last visit on May 30, 2009)
12. Trustbar Firefox addon, http://u.cs.biu.ac.il/~herzbea/TrustBar/ (Last visit on May, 30 2009)
13. Yee, K.P., Sitaker, K.: Passpet: convenient password management and phishing protection. In: SOUPS, pp. 32–43 (2006)
14. Close, T.: Petname tool 1.6, https://addons.mozilla.org/en-US/firefox/addon/957 (Last visit on May 30, 2009)
15. Drelie Gelasca, E., Tomasic, D., Ebrahimi, T.: Which Colors Best Catch Your Eyes: a Subjective Study of Color Saliency. In: Fisrt International Workshop on Video Processing and Quality Metrics for Consumer Electronics, Scottsdale, Arizona, USA (2005)
16. Wyman, B.: The persistence of identity (2006), http://www.wyman.us/main/2006/12/the_persistence.html (Last visit on May 30, 2009)

An Analysis of Widget Security

Karsten Peder Holth[1], Do van Thuan[2], Ivar Jørstad[3], and Do van Thanh[4]

[1] NTNU – O.S. Bragstadsplass 2B – N-7491 Trondheim – Norway
karsten.peder.holth@gmail.com
[2] Linus – Martin Lingesvei 17, 1330 Fornebu – Norway
t.do@linus.no
[3] Ubisafe - Gamleveien 252, 2624 Lillehammer – Norway
ivar@ubisafe.no
[4] Telenor & NTNU - Snaroyveien 30 – 1331 Fornebu - Norway
thanh-van.do@telenor.com

Abstract. Widget is a Web 2.0 concept that is gaining momentum lately. But, in order to be successful, it must have a sound security scheme. Unfortunately, until now, the security issues do not receive sufficiently attention. This paper provides a comprehensive analysis of vulnerabilities and threats for widgets. To clarify the seriousness of the threats, some known widget attacks are described. The paper proposes countermeasures to protect both the user's devices and the widget servers.

Keywords: Widget security, widget vulnerability, widget threat, Web 2.0 security.

1 Introduction

The widget concept has gained a lot of popularity in the last two years. It is considered by many Web developers as a new paradigm for access to the World Wide Web and its contents and applications. Until now, the browser has been the one and only way for entering the World Wide Web. Although it is user friendly, easy to use and flexible, it has some limitations. First, it does not provide continuous "always-on" access to services and contents. Next, for mobile users on the move and operating the mobile device quite often with one hand, it is quite a challenge to start the browser and go to the specific Web site even with the use of bookmarks.

With widgets, all the user's favourite Web sites and social communities can now be present non-stop on the desktop of the user's device. More persuasively, the user can select, instantiate, modify and delete widgets as he/she pleases. Many developers regard widgets also as a new paradigm to develop and deploy applications to the users in a ubiquitous computing environment where users are moving and using different devices on heterogeneous networks.

Indeed, the widget concept is proven to be quite compelling but in order to succeed, it is essential that widgets do not constitute a threat to either the user's devices or the widget servers. Unfortunately, so far the widget security issue has not received sufficiently attention. The goal of this paper is to shed light on the vulnerabilities and threats of the widget concept and to propose countermeasures to prevent damaging

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 60–71, 2009.

attacks. The paper starts with a short introduction to widgets. Next, the vulnerabilities of the widget system are identified. The threats are then analysed and classified. Some known attacks are also described. Countermeasures to protect both the user's devices and the widget servers are proposed. The paper concludes with suggestion for future works.

2 Brief Introduction to Widgets

Widgets are small standalone web applications usually designed for a single specific function and quick instant access to Web 2.0 services or Internet content. The World Wide Web Consortium (W3C) defines widgets as an end-user's conceptualization of an interactive single purpose application for displaying and/or updating local data or data on the Web, packaged in a way to allow a single download and installation on a user's machine or mobile device [10].

The widget may run as a standalone application, meaning outside of a Web browser, or may be embedded into a Web document. The first type of widget mentioned is categorized as a desktop or mobile widget depending on the device, and the second type is categorized as a Web widget. It is desktop and mobile widgets that are discussed in this paper.

Widgets are implemented using standard client-side web technologies such as HTML, CSS, JavaScript and XML, except that it runs in a slightly different context.

A widget engine, also known as a widget user agent, is the collection of all components needed to run widgets on the user's computer or mobile device. This includes software to handle installation and de-installation of widgets, management of widgets, as well as the user-facing functionality for invocation and configuration of the widgets [1]. When a widget is invoked, the widget user agent instantiates the Web runtime/engine, which then renders the widget on the user's device.

One of the neat parts of desktop and mobile widgets is that they can access information located on the device (e.g., memory usage, CPU temperature etc.) as well as information located on the Web, and mash these together. For instance, a weather widget can retrieve information from Weather.com and display the forecast for your area on the desktop or phonetop.

The users may download these widgets from widget portals, also known as galleries or libraries, and install them on the device. In this way the user may customize the desktop or phonetop and have access to any type of information without opening the Web browser.

3 Security Differences between Web Browsers and Widget Engines

There are fundamental differences between widgets and web applications running in Web browsers as the typical security model for widgets is closely linked to the installation and initiation process. In addition, widgets typically have access to device Application Programming Interfaces (APIs), as well as possibility to execute system commands. Traditional web applications on the other side have no installation

process; they are dynamically invoked by an URL, and run in a secure sandbox environment with security policies (e.g., same-origin policy). For the widget application model the executable content is contained within a single package persistently stored on the device; the package also contains the security restrictions as configured by the potentially malignant developer. This results in fundamental differences regarding security [11].

4 Vulnerabilities of the Widget Model

Vulnerability is a weakness in some aspect or feature of a system that makes an exploit possible. Vulnerabilities can exist at the network, host, or application levels and include operational practices [9].

In order to identify all the vulnerabilities we need to have a model of our system. In a functional view, our widget system consists of the following classes of entities:

- *User's Devices*: this entity class includes all devices belonging to all users, which can be a mobile phone or PC
- *Widget Portals*: this entity class includes all the servers hosting widgets that can be downloaded by the users. This entity class can be co-located with the widget servers.
- *Widget Servers*: this entity class include all servers which are communicating and providing content to the widgets.

If the described system is a closed one i.e. all the entities are separated from the outside and communications between entities are carried out on an internal network it is quite secured. Unfortunately, it is not the case since in between the user's device and the widget servers there is the wide open Internet which is exposed to all kinds of frauds and abuses.

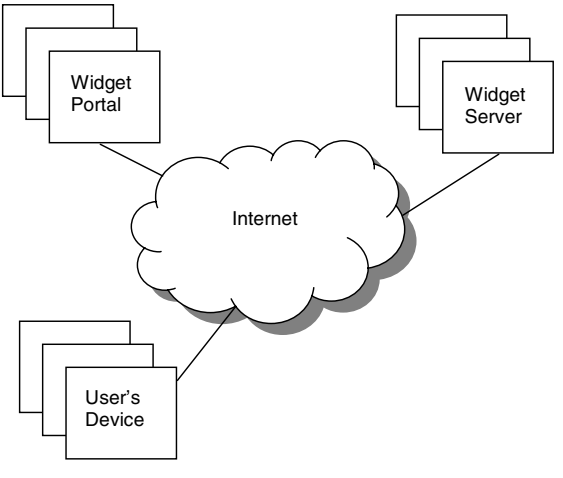


Fig. 1. A model of the widget system

The vulnerabilities of the system are composed of the vulnerabilities of each entity class of the system namely user's device, Widget Portal and Widget Server.

Vulnerabilities of the User's Device

- *The user's trust of the Widget Portal:* Most users visits Widget Portals and if they trust the portal or its owner they just simply download the widgets, instantiate and run them.
- *The lack of widget security tools*: There is no efficient tool that helps the user to verify that a widget is secure.
- *The widget engine as infiltration tunnel:* Originally, the widget engine was an isolated "sandbox" where widgets are running on but demands for more advanced widgets have led to the introduction of APIs allowing widgets to access to the device's local resources such as file system, I/O system, SIM, networking, etc. The user's device is hence more exposed to attacks.

Vulnerabilities of the Widget Server

- *The Widget Server's trust of its widgets*: By trusting the "wrong" widget, the Widget Server may execute commands which can be compromising or damaging.

Vulnerabilities of the Widget Portal

Since the Widget Portal is supposed to offer the downloading of widgets, it does not have any vulnerability as such in our model.

5 Threats

A threat is an undesired event. A potential occurrence, often best described as an effect that might damage or compromise an asset or objective. It may or may not be malicious in nature [9].

Threats on the User's Device

- *Disinformation:* Malicious widgets may supply the user with wrong information which may be quite critical in some cases like stock market, currency, trade, etc.
- *Disclosure of confidential information*: The user may be fooled by malicious widgets to reveal confidential information such as user name, password, etc.
- *Damage of resources on the user's device*: Malicious widgets may cause serious damages on the user's device, e.g. deletion or corruption of documents.
- *Access to resources on the user's device*: More seriously, malicious widgets may have access to resources and alter them.
- *Attack launch from the user's device:* Malicious widget may install and initiate the execution of malicious codes that are attacking other devices and systems.
- *Lose control over the user's device*: Ultimately, malicious widget may install and execute malicious program codes that take over full control of the user's device.

Threats on the Widget Server

- *Disclosure of confidential information*: Malicious widgets may extract information or content which is not supposed to be disclosed without contract or agreement.
- *Damage of resources on the Widget Server*: In the case of a supposed highly trusted widget, serious damage may be inflicted by the execution of high privilege methods, e.g. erasing or modifying data hosted by the Widget Server.

6 Attacks

An attack is defined in this paper as a hostile action that exploits one or more vulnerabilities of the system to materialize one of more threats on the system.

Attacks are unlimited in terms of numbers and nature. Indeed, an attack is only known when it is first detected. Here are a few attacks against widget systems that are known so far.

E-mail Subject Injection Attack

One of the most famous attacks against widget users is the e-mail subject attack found by W3C's Security Activity Lead Thomas Roessler back in 2007 [2]. This specific attack targeted an older version of the Gmail Dashboard widget, which is a frontend to the Google Mail service. The attack exploits the lack of output cleansing in the client-side JavaScript code by injecting script directly into the Document Object Model (DOM) causing it to execute the code. This is the same kind of vulnerability that is found in cross-site scripting vulnerabilities in servers. However, this vulnerability is located on the client's side. Consider the code shown below:

```
var titleText = MessagesTa-
ble.getTitleTextFromEntryElement(currentEntry);
titleText = '   <span class="title-
class">'+titleText+'</span>';
if (Prefs.getShowSnippets()) {
    var summaryText = MessagesTable.getSummary(currentEntry);
    summaryText = '<span class="snippet-class"> - ' +
    summaryText + '</span>';
    titleText += summaryText;
}
titleText = "<div class='table-overflow-col'>" + titleText +
"</div>";
...
titleColumn.innerHTML = titleText;
```

In the preceding code the non-standard innerHTML JavaScript property is used to parse a web server response. This is one of the simplest ways to do just this. However, the use of the non-standard innerHTML property to write to the DOM means that, if it is possible to inject tags into the titleText variable, it is also possible to write tags directly into that DOM. An attack could therefore be accomplished simply by sending an e-mail to the victim with tags included in the e-mail subject header. Placing script tags in the subject and get the script to execute is somewhat harder, because innerHTML does not actually execute scripts right away. However, it is possible to

bypass this problem by using event handlers, which work quite nice for the purpose. One example could be to insert the following code into the e-mail subject [2]:

```
<a href="#" onmouseover='var foo=widget.system ("curl
http://example.com/test | sh", null).outputString;'><span
class="title-class">Hello</span></a>
```

This code downloads and executes a shell script (with the user's privileges) if the user hover the mouse pointer over the subject header of the message. In other words, a computer could be seized by sending a simple e-mail, which then is triggered by a likely and innocuous user interaction [2].

Another more automatic way of triggering a potentially malicious script is by embedding an img tag, which points to an invalid URL, and use the "onerror" event to trigger the script [2]. The code would then look something like this:

```
<img src='invalidurl' onerror='evilScript()'/>
```

Malicious Content Residing in Widget Resources

The threat of malicious content cannot be stressed enough. Most users usually do not check the content of widgets after downloading it. It seems quite odd to do it in the first place. Who actually checks the source code of applications they download? They simply just install them. This opens the door for evil minded developers that prey upon exactly this trust. An attacker might create a widget which is as simple as showing the remaining days left to New Years Eve. The widget might become popular and downloaded by thousands of people from around the world. In addition, the widget is so attractive to look at that it is always opened in the widget user agent.

However, behind this innocent user interface evil code is lurking. The attacker has granted the widget network access, file access and system access. In addition to the known functionality, the widget periodically connects to a server that sends it new instructions. These instructions can be anything, ranging from a local root exploit to really gain access to the system, or delete the content of the user's documents directory.

A proof of concept has been developed, which looks for an instruction file on a server that is then downloaded and executed. It can be found at [3].

Auto-installation of Malicious Widgets without the User's Notice

The threat of auto-installation of malicious widgets without the user's notice has been known for some time, and most vendors have implemented security measures to prevent this. Before however, it was quite easy to carry through this type of attack, at least when it comes to Dashboard widgets. All that was needed was a user using Safari accessing a specific Web page. Thanks to the magic of widget auto-install, combined with a <meta> tag in the HTML file and Safari 2.0's default preference settings ("open safe files after downloading"), an evil widget gets installed on the dashboard, without the user's notice. An example of the meta tag used is provided below:

```
<meta http-equiv="refresh" content="0;
url=http://example.com/widgets/evilWidget.wdgt.zip">)
```

The meta tag states that when the page gets loaded it will refresh and redirect to the specified URL, which in this case is an evil widget resource. If the browser is configured to auto-download files the browser will associated the widget resource with the widget user agent and the installation process starts. In early versions of dashboard this installation procedure did not prompt the user for permission, but installed and instantiated the widget immediately. In other words a fully automatic attack against any Mac user was quite easy.

7 The Current Widget Security Model and Its Insufficiencies

The traditional model of delivering Web experiences to desktop or mobile users has been through Web browsers, which process content in a secure sandbox environment with limited or no possibilities of accessing native resources. However, the key benefit, and vulnerability, of Widgets is the possibility of combining Web and device features. This should require more sophisticated security mechanisms than the ones applied today, which leave all important security decisions up to developers that may have malicious intentions; developers should not be trusted with this responsibility. Most widget engines use a security block in the widgets configuration document to specify what kinds of access rights the widget shall have in the system. A typical block may look like:

```
<security>
    <http name="Example">example.com</http>
    <filesystem>sandbox</filesystem>
    <applescript>true</applescript>
    <command>true</command>
</security>
```

The example, taken from Yahoo!'s widget engine Konfabulator, illustrates what kind of access the developer may grant the widget. The only opportunity the user has to stop malicious widgets from accessing their device is by denying them to run. Most widget engines ask the user for authorization to install and instantiate the widget, this is of course a key countermeasure, but who downloads an application they do not want to install and run? There are in most cases not enough information provided to the user, so that he/she may perform a rational decision. Thus, there is no way for users to limit the possible damages without a solid security framework implemented in the widget engine, or as a common resource usable for different application platforms, that limits the malicious possibilities of the applications based on trust levels or similar criteria. It is exactly these kinds of issues that will be elaborated upon in the countermeasures part of this paper.

8 Countermeasures

Countermeasures address vulnerabilities to reduce the probability of attacks or the impacts of threats. They do not directly address threats; instead, they address the factors that define the threats. Countermeasures range from improving application design, or improving the code, to improving an operational practice [9].

Countermeasures could in this sense mean everything from good programming practices when developing both the widgets and the widget servers, as well as to integrate security into the widget engine's architecture. Security architecture goals include securing the system to prevent security breaches, minimize breaches if they occur, and react to an intrusion and recover from it if it happens.

The true difficulty lies in having a sound security model which still allows developers to create advanced, functional applications. The Software Development Kits (SDKs) should not only advice the developers, but also provide a development path where it is "easy" to both include advanced functionality but also have proper security mechanism.

If we compare Java ME and native Symbian applications for mobile, this has been the major obstacle for why these platforms never have had success – the security model is probably OK, but the development path required for properly securing applications is almost impossible to follow. Thus, it is nearly impossible to develop fancy applications.

Countermeasures that may prevent or reduce the probability of successful attacks within the widget landscape are categorized as:

- Countermeasures for widget engines
- Countermeasures for widget servers
- Countermeasures for widget developers
- Countermeasures for widget portals.

8.1 Countermeasures for Widget Engines

These countermeasures are aimed at preventing or reducing the probability of successful attacks against the devices and their users, and should be implemented in a security framework handling widgets on the user's device. They include:

- Security framework for API access to device capabilities
- Digital signature program and author integrity check
- Cross-domain policy files
- User notification
- Client-side whitelists
- Input sanitation.

Security Framework for API Access to Device Capabilities

The JavaScript APIs access to device capabilities need to be controlled by a security framework in order to mitigate the risks created by allowing widgets access to device capabilities. The high-level objective of such a security framework would be to protect the device and its user from widgets that pose a security risk from both unintentional implementation flaws and malicious content.

Digital Signature Program and Author Integrity Check

Digital signature programs aim to allow legitimate developers to contribute applications while keeping viruses, malware, and privacy attacks under control. It is also independent of distribution method, because the digital signature is not linked to the specific widget portal, but to the actual author.

The approach can be used to distinguish trust level at which the signed widget can run. One example of such an approach can be found in the iPhone platform, which requires code to be signed twice (first with developer certificate, secondly with App-Store certificate) before it can be sold in Apple's AppStore. Another example is the Symbian signed initiative.

A widget should not necessarily need to be signed to be able to run, but a signature should be required when access to certain capabilities are requested. In this way it is possible to enforce different trust levels at which the widget can run.

Arguments against using digital signatures mainly state that it hinders the potential for a truly widely accepted open mobile application platform. However, the argument has received some adverse press attention due to virus threats (namely Trojan horses) [8].

Cross-domain Policy Files

A cross-domain policy file is a simple XML file that specifies permission to access a given domain or device feature. These files are specified by potentially malignant developers and should only be used in combination with a digital signature program as the one described in this chapter. Most widget vendors have chosen exactly this strategy, but without the signature program.

User Notification

Informing the user about the access privileges of the Widget should be done whenever the widget gets installed, as well as every time it gets instantiated. The current practice in this field is quite alarming. Although it has become standard practice to prompt the user about permission to instantiate the widget under the installation process, many frameworks do not provide the users with detailed information about what kinds of dangers they might expose themselves to. One of the vendors that provide more detailed information to the users is Yahoo!. They provide the user with a drop down list of all the security critical actions the widget may use. This may be information regarding restrictions to access the Internet or the file system, as well as a list of applications that can be run by the widget. However, not all users have the competence to evaluate what is safe or not. It might not occur to the user that "unrestricted access" means access to anything on the computer, including the user's personal files. The warning should clearly specify what kind of dangers the widget might contain. The difficulty is to find a balance between usability and security.

Client-side Whitelists

Rather than blacklisting all unwanted functionality, it is possible to instead whitelist what the widget should be able to do. In this way everything that is not on the white list is unsupported. This may include the ability to delete files etc.

Input Sanitation

Neutralize script injection attempts by whitelisting known markup that should be supported, rather than blacklisting specific known pattern for XSS attacks. One example could be to initially neutralize HTML tags, but afterwards turn any <b&gt (character entity) back into a <b>.

8.2 Countermeasures for Widget Servers

These countermeasures are aimed at preventing or reducing the probability of successful attacks against the Widget server. They include:

- Validate all XmlHttpRequests from the user's device
- Deny actions by default; use whitelists to authorize actions

In practice, many security problems occur simply because the Widget/Web server administrators assume that users behave properly and play by the rules. This means that they often assume that users do not modify the widget's scripts to acquire different information from the service. However, in a secure system the security controls are implemented in an environment which is outside the control of the end user.

8.3 Countermeasures for Widget Developers

These countermeasures are aimed at preventing or reducing the probability of successful attacks against devices or their users. They include:

- Digitally signing the widget resource
- Good programming practises
- Limiting the widget's access to the bare minimum
- Validate input from the widget server.

Digitally Signing the Widget Resource

By signing the widget resource the author can be authenticated and the integrity of the widget can be verified. Digital signature programs aims to allow legitimate developers to contribute applications while keeping viruses, malware, and privacy attacks under control. Please see the countermeasure for widget engines section for further information.

Good Programming Practises

One of the most obvious countermeasures would be good programming practices for widgets. It has become clear that fancy security models are not enough. Developers need to be aware of the dangers they can expose their widget users to. A classical piece of bad programming is parsing data by evaluating it as code. This has regrettably shown to be common practice among widget authors, as seen in the e-mail subject injection attack presented in the attack chapter [12]. However, developers often do this to save time. It is therefore important to inform developers, either by discussing it in forums, making papers like this, or adding advice/guidelines in the documentation provided in SDKs.

Limiting the Widget's Access to the Bare Minimum

This includes specifying exactly which web domains and which device features shall be accessed by the widget. Coarse grained specification of access to Internet and device features is not recommended, for example, "allow device access" should instead be specified more fine grained, i.e. "allow access to SMS functionality".

Validate Input from the Widget Server

By validating the input from the widget server before processing it further the probability of a successful attack from a malicious server should decrease.

8.4 Countermeasures for Widget Portals

The staff behind the different widget portals should scan their widget libraries for malicious widgets and provide functionality that enables users to notify the staff about strange behaviour after installing the widget. This countermeasure is aimed at preventing or reducing the probability of successful attacks against user devices and their users. The serious widget vendors do in most cases check the uploaded widgets for errors or malicious content and some even check widgets already published, especially those that are quite popular. This is the first obstacle evil-minded developers need to circumvent when trying to get their malicious widgets out to the masses. Reports from Google have however revealed that there is rarely found anything at all.

9 Conclusions

In this paper, the security issue of widgets has been considered. The vulnerabilities of the widget model and the related threats have been analysed thoroughly. Some known attacks have been described. The main contribution of the paper is the complete analysis of widget security, as well as the proposition of countermeasures to protect the widget system. However, it is worth mentioning that the strength of widgets lies on their ease of use and flexibility which contradict security measures. Any exaggeration of protection could be a hindrance for the success of widgets. For future works, it may be necessary to elaborate the widget security framework further. The classification of widgets should be carried out and access rights to device APIs must be defined. Only access to critical resources will require higher level of clearance.

References

1. BONDI Architecture & Security Requirements version 1.0. OMTP (February 9, 2009)
2. Roessler, T.: More on widgets: When one e-mail is enough to break a system, Thomas Roessler's blog (December 19, 2007), http://log.does-not-exist.org/archives/2007/12/19/2159_more_on_widgets_when_one_email_is_enough_to_break_a_system.html
3. Apple Dashboard Widget Insecurity, Geisterstunde.org (October 6, 2008), http://www.geisterstunde.org/drupal/?q=node/67
4. Access Control for Cross-Site Requests, W3C Working Draft (September 12, 2008)
5. Schuh, J.: Same-Origin Policy Part 1: Why we're stuck with things like XSS and XSRF/CSRF (February 8, 2007), http://taossa.com/index.php/2007/02/08/same-origin-policy
6. Zalewski, M.: Browser Security Handbook, part 2, Google Inc. (2008), http://code.google.com/p/browsersec/wiki/Part2
7. Symbian: How do I get my Symbian OS application signed? https://www.symbiansigned.com/how_do_I_get_my_application_signed_2.5.pdf

8. Thomson, L.: Mobile virus moves to new level – when friends infect (April 5, 2005), http://www.vnunet.com/vnunet/news/2127090/mobile-virus-moves-level
9. Maier, J.D., Mackman, A., Wastell, B.: Threat Modeling Web Applications, Patterns & Practices Library, Microsoft Corporation (2005)
10. Widgets 1.0: The Widget Landscape, W3C Working Draft (April 14, 2008)
11. Liwell, M., Nilsson, C.: Sony Ericsson Position Paper for the W3C Workshop Security for Access to Device APIs from the Web in London (December 10-11, 2008), http://www.w3.org/2008/security-ws/papers/SEMC_Position_Paper.pdf
12. Roessler, T.: When Widgets Go Wrong, W3C Q&A Blog (December 20, 2007), Available at http://www.w3.org/QA/2007/12/when_widgets_go_wrong.html

Trade-Offs in Cryptographic Implementations of Temporal Access Control

Jason Crampton

Information Security Group, Royal Holloway, University of London
jason.crampton@rhul.ac.uk

Abstract. In recent years, we have seen the development of key assignment schemes that use cryptography to enforce time-based authorization policies. One of the most important aspects of these schemes is the balance between the time required to derive keys and the amount of storage required for the public information from which keys are derived. The derivation time and storage are dependent on the number of time periods used in the authorization policy. In this paper, we discuss novel schemes that achieve good trade-offs between these competing parameters and for which explicit bounds can be given in terms of the number of time periods.

1 Motivation

In some situations, we may wish to use cryptographic techniques to implement some form of access control. By (symmetrically) encrypting the contents of a data object, we restrict access to an object to those users who possess the corresponding encryption key. The problem we now need to solve is the efficient and accurate distribution of encryption keys to authorized users.

In recent years, there has been a considerable amount of interest in key encrypting schemes. In such schemes, a user is given a secret value – typically a single key – which enables the user to derive some collection of encryption keys. Key derivation is performed using the secret value and some information made publicly available by the scheme administrator.

The most widely studied situation assumes the existence of a partially ordered set $(L, \leqslant)$ and an encryption key $\kappa(x)$ associated with each element $x \in L$. Each user is associated with a security label $\lambda(u) \in L$ and should be able to derive all keys in the set $\{\kappa(y) : y \leqslant \lambda(u)\}$. This scenario is based on the information flow policies for confidentiality that were widely studied in the 1970s [5,12]. It is generally assumed that the trivial solution in which the user's secret is the set of keys $\{\kappa(y) : y \leqslant \lambda(u)\}$ is not of interest. Instead, the research literature has focused on so-called *key assignment schemes* in which each user is given a single key and additional public information is used to derive additional keys. The two objectives when designing such a scheme are to minimize the amount of public information and the time required to derive a key. Unsurprisingly, it is not possible to realize both objectives simultaneously, so trade-offs have been

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 72–87, 2009.

sought. Crampton *et al* provide a survey of, and taxonomy for, key assignment schemes, and the various factors that affect the parameters described above [9].

At the same time, we have seen the development of access control models in which time plays an important role in deciding whether access requests are authorized or not [6]. One particular application of such "temporal access control" systems is the protection of data that is made available periodically as (part of) a subscription-based service [7].

Crampton *et al* suggested that temporal access control policies, in which there are number of time points and users were authorized to access data over a sequence of consecutive time points, could be implemented using key assignment schemes. Shortly after, Atallah *et al* [2], Ateniese *et al* [4] and De Santis *et al* [11] described key assignment schemes for temporal access control using this representation for time. Their work focused on two particular aspects:

- the development of schemes that were secure against key recovery, and
- the reduction of the storage required for public information and the number of operations required for key derivation.

One shortcoming of their work is that the approaches used to tackle the second of these issues do not consider the actual requirements of the underlying access control model. Instead, generic techniques to reduce the diameter of a directed graph are applied. This has two consequences: application-specific optimizations are not considered and only the asymptotic behavior of the constructions is provided. Given that the number of time intervals is likely to be rather small in many practical applications, it is not clear that this kind of approach is the most appropriate.

In this paper, we consider application-specific optimizations that arise from two rather straightforward observations. This enables us to present concrete schemes with precise bounds on the amount of storage and the number of derivation steps required. One nice feature of this work is that we may still be able to exploit the techniques used in earlier work to obtain some further reduction in storage and derivation time.

In the next section, we describe some relevant background material. We then make a simple, yet powerful, observation and explore the schemes that can be derived as result of this observation. In Sect. 4, we explore the consequences of another simplification. Section 5 describes a different way of decomposing the partially ordered set of time intervals, which yields methods for fixing the number of derivation steps required. We conclude the paper with a summary of our contributions, a comparison with related work and some suggestions for future work.

2 Key Assignment Schemes

Let us suppose we have a partially ordered set of security labels $(L, \leqslant)$. An information flow policy requires that each user u and protected object o be assigned a security label; u is authorized to read o provided the security label

of u is greater than or equal to that of o. More formally, let $\lambda : U \cup O \rightarrow L$ be a labeling function that associates each entity with a security label. Then u is authorized to access o if and only if $\lambda(u) \geqslant \lambda(o)$.

A *key assignment scheme* may be used to enforce an information flow policy. A key assignment scheme comprises a set of keys $\{\kappa(x) : x \in L\}$ and a set of public information. Each object with security label x is encrypted with $\kappa(x)$. A user u with the key $\kappa(\lambda(u))$ must be able to derive $k(y)$ for any $y \leqslant \lambda(u)$ using $\kappa(\lambda(u))$ and public information. Hence, a user can decrypt any object with security label $\lambda(y)$, where $y \leqslant \lambda(u)$. The parameters that characterize the behavior of a key assignment scheme are [9]:

- the number of keys that a user requires;
- the amount of public information that is required;
- the amount of time taken to derive a key.

Generally speaking, we are only interested in schemes in which each user has a single key. (We could, trivially, give a user a key for each of the labels that are less than or equal to $\lambda(u)$ – that is, the set of keys $\{\kappa(y) : y \leqslant \lambda(u)\}$ – but this type of approach is rarely considered appropriate.) As a consequence we require either more public information or longer key derivation times (or both).

2.1 Key Assignment Schemes for Directed Graphs

A partially ordered set $(L, \leqslant)$ can be represented by a graph (L, E). There are two obvious choices for the edge set E: one is the full partial order relation $\leqslant$; the second option is to omit all transitive and reflexive edges from $\leqslant$ to obtain the *covering relation*, denoted $\lessdot$. The graph $(L, \lessdot)$ is called the *Hasse diagram* of L, and is the usual representation of L as a directed graph [10].

Note that we can easily generate a key assignment scheme for any directed acyclic graph (V, E). Specifically, we associate a key $\kappa(x)$ with each $x \in V$, and, for each edge $(x, y) \in E$, we publish $Enc_{\kappa(x)}(\kappa(y))$, where $Enc_k(m)$ denotes the encryption of message m using key k.[1] Then any user in possession of $\kappa(x)$ can derive $\kappa(y)$ in one step, and for any z on a path from x containing m edges, $\kappa(z)$ can be (iteratively) derived in m steps.

2.2 Trade-Offs in Key Assignment Schemes

It can seen that key assignments schemes for directed graphs can be used specifically for information flow policies. We may use the graph $(L, \leqslant)$, in which case key derivation can always be performed in one step. In contrast, key derivation may require a number of steps if we use the graph $(L, \lessdot)$. The trade-off here is that the second graph contains fewer edges and hence the number of items of public information that are required to support key derivation is smaller. The study of these kinds of trade-offs will be the focus of this paper.

[1] Atallah *et al*, for example, use the "encryption" $\kappa(y) \oplus h(\kappa(x) \parallel y)$, where h is a hash function, $\oplus$ is bit-wise XOR, and $\parallel$ denotes concatenation of strings [3].

2.3 Security Considerations

A number of schemes exist in the literature that prevent a user from obtaining keys for which she is not authorized by colluding with other users. The first such scheme was developed by Akl and Taylor [1] specifically for partially ordered sets. Since then Atallah *et al* [2] and, independently, Ateniese *et al* [4] have shown that there exist key assignment schemes that are secure against "key recovery" and can be used with arbitrary directed graphs. We will assume henceforth that the schemes we propose, which only vary in the choice of edge set for the graph, will be implemented using a scheme that is secure against key recovery.

2.4 Key Assignment Schemes for Temporal Access Control

Let us now assume that users are authorized to access objects for specified periods of time. We assume that there exists a sequence of time "points" $t_1, \ldots, t_m$ and that there are cryptographic keys $\kappa_1, \ldots, \kappa_m$ associated with each time point. Each key κ_i is used to encrypt data (using some appropriate symmetric encryption technique) released at time point i. Each time point is a collection of time instants; that is, a time interval. A sequence of consecutive time points represents a longer time interval.

We can use a key assignment scheme to enforce a temporal access control scheme. Specifically, the lattice of security labels is the set of intervals defined over the set $\{t_1, \ldots, t_m\}$ ordered by subset inclusion, which we denote by I_m. That is, $I_m = \{[t_i, t_j] : 1 \leqslant i \leqslant j \leqslant m\}$, where $[t_i, t_j]$ denotes the set $\{t_i, t_{i+1}, \ldots, t_{j-1}, t_j\}$. Clearly, we may represent $\{t_1, \ldots, t_m\}$ simply as the set $\{1, \ldots, m\}$, which we denote by $[m]$. Henceforth, we use this representation of, and notation for, the set of time points: a user authorized for time points $i, \ldots, j$ is assigned to label $[i, j]$; key (object) κ_i is assigned to label $[i, i]$.

Note that the number of edges in the graph $(I_m, \preccurlyeq)$ contains $m(m-1)$ edges. Note also that the Hasse diagram of I_m can be represented as a triangular grid, with the apex of the triangle representing the interval $[1, m]$ and the long diagonal representing the intervals $[i, i]$.[2] This graphical interpretation of I_4 is illustrated in Fig. 3(a) on page 78. Using this graph, we can see that iterative key derivation requires precisely $m - 1$ steps to get from vertex $[1, m]$ to any vertex of the form $[i, i]$. More generally, it requires $j - i$ steps to get from the vertex $[i, j]$ to a vertex $[l, l]$, where $l \in [i, j]$.

Henceforth, we will call the triangular pattern of side m an *m-triangle*, denoted by T_m. We will write D_m to denote an *m-diamond* – a diamond-shaped pattern of side m (obtained by joining two copies of T_m along the long diagonal). T_4 and D_4 are illustrated in Fig. 1. Nodes of the form $[i, i] \in T_m$ will be called *leaf nodes*.

To conclude this section, we summarize the problem we address in the remainder of the paper. We wish to construct a graph (T_m, E_m), such that

- for any non-leaf node $[i, j]$ there exists a path from $[i, j]$ to $[k, k]$ for every $k \in [i, j]$;

[2] All edges in all graphs depicted in this paper are assumed to point down the page.

(a) T_4 (b) D_4

Fig. 1. T_4 and D_4

– $|E_m|$ is small;
– the length of the longest path in (T_m, E_m) is small.

The first criterion simply reflects that the graph must implement the desired access control policy. The second means that we wish to keep the public storage requirements small, while the final criterion means that we wish to make the worst-case key derivation time small. Note that T_m is fixed, so any scheme is defined by the choice of E_m. For brevity, we will simply refer to a scheme E_m (with T_m and the graph (T_m, E_m) left implicit).

3 An Immediate Improvement

We start by making the first of our application-specific observations. This immediately yields a number of significant improvements to the number of steps required.

Remark 1. *The only security labels for which a user needs to derive a key have the form $[i, i]$ (since no objects are associated with any interval of the form $[i, j]$, where $i < j$). Therefore, it is irrelevant whether a user with security label $[i, j]$ can derive $[i', j']$, where $[i', j'] \subseteq [i, j]$ and $i' \neq j'$.*

This means that we can seek to optimize the key assignment scheme used to derive keys by omitting some of the edges in the triangular grid I_m. Clearly, the triangular grid is "self-similar", so it is natural to examine recursive methods for generating graphs in which the number of hops is reduced. Before we discuss such methods, we introduce a very simple 1-hop scheme.

Scheme 1 (1-hop derivation). *In our first scheme $E_m^{(1)}$, we simply insert an edge between every non-leaf node $[i, j]$ to leaf node $[k, k]$, for all $k \in [i, j]$.*

Scheme $E_4^{(1)}$ is illustrated in Fig. 2. We can see that 16 edges are required: 4 for the top node, 3 for each of the nodes in the second row, and 2 for each of the nodes in the third row.

Proposition 2. *For all $m \geqslant 1$, $e_m^{(1)} = \frac{1}{6}m(m-1)(m+4)$, where $e_m^{(1)}$ denotes the cardinality of $E_m^{(1)}$.*

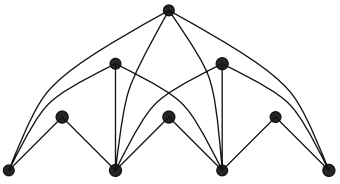


Fig. 2. The scheme $E_4^{(1)}$

Proof. It is easy to see that we require m edges for the uppermost node, $m-1$ edges for each of the two nodes in the next row, etc. Hence, we see that the number of edges required is given by

$$\begin{aligned}\sum_{i=1}^{n-1} i(n-i+1) &= (n+1)\sum_{i=1}^{n-1} i - \sum_{i=1}^{n-1} i^2 \\ &= \frac{1}{2}(n+1)(n-1)n - \frac{1}{6}n(n-1)(2n-1) \\ &= \frac{1}{6}n(n-1)(3n+3-(2n-1)) \\ &= \frac{1}{6}n(n-1)(n+4)\end{aligned}$$

□

Scheme 2 ($\log_2 m$-hop derivation). *To construct our second scheme $E_m^{(2)}$, we make the following observations:*

- *T_{2m} contains two copies of T_m and one copy of D_m;*
- *no edges are required between nodes in D_m (by Remark 1);*
- *I_m contains two edges for each node in D_m*
- *for each node $[i,j] \in D_m$, $m, m+1 \in [i,j]$.*

Hence, to construct $E_m^{(2)}$, we replace the two edges in I_m from each node $[i,j] \in D_m$ with two edges $([i,j],[i,m])$ and $([i,j],[m+1,j])$. In other words, each node in D_m is directly connected to a node in each copy of T_m.

Note that the number of derivation hops for T_{2m} is one more than that for T_m. Henceforth, we will write $h_m^{(i)}$ for the number of derivation hops in scheme $E_m^{(i)}$. In particular, we have $h_{2m}^{(2)} = h_m^{(2)} + 1$.

Figure 3(b) illustrates $E_4^{(2)}$. Note that it is not possible to get from the top node to a leaf node in 1 hop. Note, however, that only 12 edges are required.

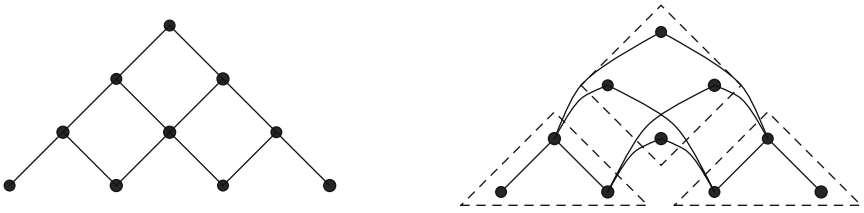
In an effort to keep the notation simple, we will write h_m and e_m (rather than $h_m^{(i)}$ and $e_m^{(i)}$) where no confusion can arise about which scheme is under consideration.

Proposition 3. *For scheme $E_n^{(2)}$, $n \geqslant 2$, we have*

$$e_{2n} = 2e_n + 2n^2 \tag{1}$$

$$e_n = n(n-1) \tag{2}$$

$$h_n = \log_2 n \tag{3}$$



(a) I_4 (b) $E_4^{(2)}$

Fig. 3. Constructing the scheme $E_4^{(2)}$

Proof. The first result follows immediately from the construction. Assuming we use scheme $E_n^{(2)}$, then we need two copies of T_n each containing e_n edges and we need two edges for each of the n^2 nodes in D_n.

To prove (2), recall that I_n contains $n(n-1)$ edges and note that the construction of $E_n^{(2)}$ replaces the two edges for each node in D_n with two different edges. Hence, e_n is the same as the number of edges in I_n.

Finally, note that $h_{2n} = h_n + 1$. Solving this recurrence relation, we obtain $h_n = \log_2 n$. □

The important thing to note about $E_n^{(2)}$ is that it requires no more storage than I_n, but the upper bound on key derivation is $\log_2 n$ steps rather than $n-1$ steps. Finally, we note that our approach can also be applied to triangle T_{m+n}. We illustrate the scheme $E_5^{(2)}$ in Fig. 4.

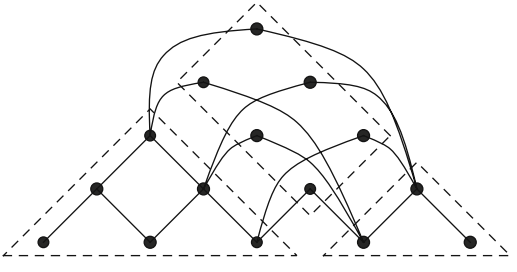


Fig. 4. Constructing $E_5^{(2)}$ using T_2 and T_3

4 Skipping a Level

Suppose we have a scheme for T_n which requires no more than h_n steps, and we have used Scheme 2 and the scheme for T_n to construct a scheme for T_{2n} with $h_n + 1$ steps. Now consider T_{4n}, which breaks down into two copies of T_{2n} and one copy of D_{2n}. Note that D_{2n} can be sub-divided into four copies of D_n. Nodes

in each copy of D_n can then be attached to nodes in the appropriate copies of T_n contained within each T_{2n}, skipping the intermediate nodes in T_{2n}. Hence, we do not increase the total number of hops required for derivation: a node in D_{2n} is connected directly to a node in T_n, as are nodes in each copy of T_{2n}. In other words, we have $h_{4n} = h_n + 1 = h_{2n}$. More formally, we have the following construction.

Scheme 3. *Let $n = bm$. Then for T_{2n}, we split D_n into b^2 copies of D_m. We connect each node in each D_m to the appropriate nodes in copies of T_m. In particular, for $[i, j]$, where $i = \alpha_i m + \beta_i$ and $j = \alpha_j m + \beta_j$, we add edges from $[i, j]$ to nodes $[i, (\alpha_i + 1)m], [(\alpha_i + 1)m + 1, (\alpha_i + 2)m], \ldots, [\alpha_j m, j]$.*

The (partial) construction of $E_8^{(3)}$ is illustrated in Fig. 5. Node $[2, 6]$ in D_4 has been connected to nodes $[2, 2]$, $[3, 4]$ and $[5, 6]$. Note that $[2, 6]$ would have been connected to nodes $[2, 4]$ and $[5, 6]$ in $E_8^{(2)}$. Note also that $h_8^{(3)} = h_4^{(3)} = h_2^{(3)} + 1$.

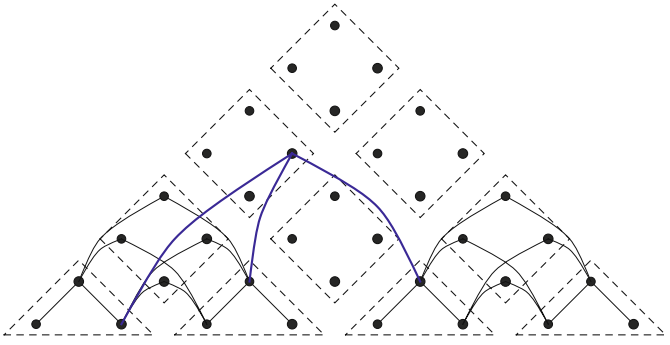


Fig. 5. The partial construction of $E_8^{(3)}$

Lemma 4. *For scheme $E_n^{(3)}$, where $n = bm$, we have*

$$e_{2n} = 2e_n + (b + 1)n^2.$$

Moreover, if $h_n > h_m$ then $h_{2n} = h_n$.

Proof. Omitted due to space constraints.

Notice the trivial case $b = 1$ corresponds to $e_{2n} = 2e_n + 2n^2$. Of course, in this case we add a step to the derivation cost, because we do not skip a level.

We now apply this construction to the case $b = 2$, in doing so deriving a scheme $E_m^{(4)}$ in which $h_n = \frac{1}{2} \log_2 n$. We define schemes for T_2 and T_4, each of which use one derivation step. Then we can define a scheme for T_8 that requires two derivation steps using two copies of T_4 as the building blocks. We can define a scheme for T_{16} that also requires two derivation steps using four copies of T_4 as the building blocks. Extending further, we define a scheme for T_{32} that

Table 1. Constructing a scheme with $\frac{1}{2}\log n$ derivation steps

n	Steps	Derivation of e_n	e_n
2	1		2
4	1		16
8	2	$2e_4 + (2^0+1).4^2$	64
16	2	$2e_8 + (2^1+1).8^2$	320
32	3	$2e_{16} + (2^0+1).16^2$	1152
64	3	$2e_{32} + (2^1+1).32^2$	5376
128	4	$2e_{64} + (2^0+1).64^2$	18944
256	4	$2e_{128} + (2^1+1).128^2$	87040

requires three derivation steps using two copies of T_{16} as the building blocks, and define a scheme for T_{64} that also requires three steps using four copies of T_{16} as building blocks, etc. This construction is illustrated in Table 1. Using the result of Lemma 4 and noting that $b = 1$ or $b = 2$ in this construction, we have $e_{2n} \leqslant 2e_n + 3n^2$.

Lemma 5. *For $E_n^{(4)}$ and $n = 2^m$, we have $e_n \leqslant \frac{3}{2}n(n-1)$.*

Proof. We prove the result by induction on m. We choose a 1-hop scheme for T_4 with 16 edges. Hence, the result holds for $m = 2$. Now let us assume that the result holds for $m \leqslant M$ and consider $e_{2^{M+1}}$. Then, by construction,

$$\begin{aligned} e_{2^{M+1}} &\leqslant 2e_{2^M} + \frac{3}{2}2^{2M} \\ &\leqslant 2\left(\frac{3}{2}(2^{2M} - 2^M)\right) + \frac{3}{2}2^{2M} \qquad \text{(by inductive hypothesis)} \\ &= \frac{3}{2}\left(3.2^{2M} - 2^{M+1}\right) < \frac{3}{2}\left(2^{2M+2} - 2^{M+1}\right) = \frac{3}{2}2^{M+1}(2^{M+1} - 1) \end{aligned}$$

as required. □

We conjecture that we can use $b = 2^r$, for any non-zero positive integer r, to derive a scheme with $h_n \approx \frac{1}{r+1}\log_2 n$ and $e_n \leqslant an(n-1)$, where a is a function of r. The sole obstacle to the direct construction of a scheme analogous to $E_n^{(4)}$ is the construction of a scheme for the base case T_{2^r} such that $e_{2^r} \leqslant \frac{1}{2}a2^r(2^r - 1)$. In particular, we will not be able to use a 1-hop scheme for T_{2^r} (unlike the case $b = 2$), because the number of edges in a 1-hop scheme for T_n grows with the cube of n. However, it should always be possible to find a scheme for T_{2^r} with a small number of hops so that the base case "works". This will be the subject of future work.

We also note that we can construct a scheme that only requires $\log_2(\log_2 n)$ derivation steps. We define schemes for T_2 and T_4, each requiring a single derivation step. Then for $T_8 = T_{2^3}$ and $T_{16} = T_{2^4}$ we build schemes using $T_4 = T_{2^2}$.

For $T_{2^5}, \ldots, T_{2^8}$, we build schemes using T_{2^4}. For $T_{2^9}, \ldots, T_{2^{16}}$, we build schemes using T_{2^8}. This construction is illustrated in Table 2. A similar technique can be applied for any $n = 2^{2^k}$. In particular, we have the following result.

Lemma 6. *Let $n = 2^{2^k}$. Then we can construct a scheme for T_n such that*

$$e_n - \sqrt{n}e_{\sqrt{n}} = \frac{1}{6}n\sqrt{n}(\sqrt{n}-1)(\sqrt{n}+4) \quad \textit{and} \quad h_n = \log_2(\log_2 n).$$

Proof. Omitted due to space constraints.

Using this formula, we have, for example,

$$e_{256} = 16e_{16} + \frac{1}{6}.256.16.15.20 = 209920,$$

which can be confirmed by Table 2. Clearly, the number of edges required is approximately $\frac{1}{6}n^2\sqrt{n}$, since the term in $n^2\sqrt{n}$ will dominate the term in $\sqrt{n}e_{\sqrt{n}}$.

Table 2. Constructing a scheme with $\lceil \log_2(\log_2 n) \rceil$ derivation steps

n	$\log_2 n$	$\lceil \log_2(\log_2 n) \rceil$	Derivation of e_n	e_n
2	1	1		2
4	2	1		16
8	3	2	$2e_4 + (2^0+1).4^2$	64
16	4	2	$2e_8 + (2^1+1).8^2$	320
32	5	3	$2e_{16} + (2^0+1).16^2$	1152
64	6	3	$2e_{32} + (2^1+1).32^2$	5376
128	7	3	$2e_{64} + (2^2+1).64^2$	31232
256	8	3	$2e_{128} + (2^3+1).128^2$	209920
512	9	4	$2e_{256} + (2^0+1).256^2$	550912
1024	10	4	$2e_{512} + (2^1+1).512^2$	1888256
2048	11	4	$2e_{1024} + (2^2+1).1024^2$	9019392
4096	12	4	$2e_{2048} + (2^3+1).2048^2$	55787520
8192	13	4	$2e_{4096} + (2^4+1).4096^2$	396787712
16384	14	4	$2e_{8192} + (2^5+1).8192^2$	3008167936
32768	15	4	$2e_{16384} + (2^6+1).16384^2$	23464640512
65536	16	4	$2e_{32768} + (2^7+1).32768^2$	185441976320

5 A Multiplicative Decomposition of T_n

Let $n = ab$. Then we can split T_n into b copies of T_a and t_{b-1} copies of D_a, where $t_k = \frac{1}{2}k(k+1)$ is the kth triangle number. We treat the copies of D_a and T_a as "supernodes" in T_b. We can construct schemes for T_a and T_b and use a^2 copies

(a) T_{12} (b) T_3 with T_4 and D_4 as supernodes

Fig. 6. Creating a scheme for T_{12} using schemes for T_3 and T_4

of the scheme (one for each node in the non-leaf supernode) for T_b to connect the set of D_as to the T_as. We then use the scheme for T_a to derive keys within T_a. The construction is illustrated in Fig. 6 for T_{12}.

The simplest case arises when $b = 2$ and gives rise to Scheme 2. In this case, we have two copies of T_a and one copy of D_a. We use the obvious 1-hop scheme for T_2. We then recursively apply this construction to T_a. This process ultimately yields a scheme which requires no more than $\log_2 n$ steps.

5.1 A 2-Hop Scheme

Let us now consider the case where we would like to construct a scheme that requires no more than 2 steps. Then we need 1-hop schemes for T_a and T_b. Scheme 1 may be used to create such schemes requiring

$$\frac{1}{6}a(a-1)(a+4) \quad \text{and} \quad \frac{1}{6}b(b-1)(b+4)$$

edges, respectively. Hence, we can construct a scheme for T_{ab} that requires no more than 2 steps, and the number of edges is

$$a^2\frac{1}{6}b(b-1)(b+4) + b\frac{1}{6}a(a-1)(a+4) = \frac{1}{6}ab(a(b-1)(b+4) + (a-1)(a+4)).$$

Returning to the construction of a 2-hop scheme for T_{12} as illustrated in Fig. 6, we divide T_{12} into copies of D_4 and T_4, yielding a copy of T_3 in which the non-leaf supernodes are diamonds and leaf supernodes are triangles. We then construct a 1-hop scheme for T_3 using 6 edges (Fig. 6(b)). We duplicate this scheme for every node in the root supernode. Clearly, there are $4^2 = 16$ nodes in each supernode. Hence we require 6.16 such edges. Having done this, we can now get from any node that is contained in a copy of D_4 to a node in T_4 in one hop. It remains, therefore, to construct a scheme for each T_4 supernode such that we can get from any non-leaf node to a leaf node in one hop. Again, we can use Scheme 1 for this (as illustrated in Fig. 2). We require 16 edges for each of the three copies of T_4. The scheme therefore requires a total of $16.6 + 3.16 = 16.9 = 144$ edges.

5.2 Minimizing the Number of Edges in the 2-Hop Scheme

The obvious question to consider is how we choose a and b so that the total number of edges e is minimized. The number of edges e is given by the formula

$$\begin{aligned} e &= \frac{1}{6}ab(a(b-1)(b+4)+(a-1)(a+4)) \\ &= \frac{1}{6}ab(ab^2+3ab-a+a^2-4) \end{aligned}$$

Writing $b = n/a$, we obtain e as a function of n and a:

$$e = \frac{1}{6}n\left(\frac{n^2}{a}+3n-a+a^2-4\right)$$

Differentiating e with respect to a we obtain

$$\frac{1}{6}n\left(-\frac{n^2}{a^2}-1+2a\right) = \frac{1}{6}n(-b^2-1+2a)$$

This expression evaluates to 0 when $2a = b^2+1$. Hence, the minimum number of edges occurs when $2a = b^2+1$. Table 3 shows how the number of edges varies for $n = 12$. (The values of $2a$ and b^2+1 are tabulated for convenience.) We see that the smallest number of edges occurs when $a = 4$. In this case, $b = 3$ and $2a \approx b^2+1$. Notice that $a = 1$ and $a = 12$ are boundary cases in which the whole scheme requires one hop (since any scheme for T_1 requires 0 hops). Hence the combined scheme is equivalent to a simple 1-hop scheme.

Let us now assume that $n = ab$ and $a = \frac{1}{2}(b^2+1)$. (In other words, we have chosen n so that the minimum number of edges will be attained.) Substituting $a = \frac{1}{2}(b^2+1)$ into e, we obtain

$$\begin{aligned} e &= \frac{1}{24}n(3b^4+6b^3+2b^2+6b-17) \\ &= \frac{1}{24}n\left(3b^2(b^2+1)+6b(b^2+1)-b^2-17\right) \end{aligned}$$

$$\begin{aligned} &\leqslant \frac{1}{24}n\left(6bn+12n\right) \qquad \text{since } n = ab = \frac{1}{2}b(b^2+1) \\ &= \frac{1}{4}n^2(b+2) \end{aligned}$$

Therefore, when $n = \frac{1}{2}b(b^2+1)$, there exists a 2-hop scheme for T_n with no more than $\frac{1}{4}n^2(b+2)$ edges. When $b = 3$, for example, we have $a = 5$ and $n = 15$. In other words, the best 2-hop scheme for T_{15} arises when we use copies of T_5 and D_5 as supernodes and use a 1-hop scheme for T_3 to move between the supernodes; for this construction, we have $e = 265$. The upper bound derived above is $\frac{1}{4}15.15.5 = \frac{1125}{4}$ which clearly exceeds 265.

Table 3. How the number of edges varies for different 2-hop schemes for T_{12}

a	b	Edges	$2a$	b^2+1
1	12	352	2	145
2	6	212	4	37
3	4	172	6	17
4	3	160	8	10
6	2	172	12	5
12	1	352	24	2

Since $n = \frac{1}{2}b(b^2+1)$, we have $b < \sqrt[3]{2n}$. Notice that a 1-hop scheme requires $\frac{1}{6}n(n-1)(n+4)$ edges – that is, approximately $\frac{1}{6}n^2(n+4)$ edges – while a 2-hop scheme requires approximately $\frac{1}{4}n^2(\sqrt[3]{2n}+2)$. The 1-hop scheme for T_{15}, for example, requires 665 edges, compared to 265 for the 2-hop scheme described in the previous paragraph.

Note that $\sqrt{n}$, while perhaps being the natural choice for a (and b), is not the optimal choice. To see this, note that the number of edges e for such a 2-hop scheme is given by

$$\begin{aligned} e &= \frac{1}{6}n(\sqrt{n}(\sqrt{n}-1)(\sqrt{n}+4) + (\sqrt{n}-1)(\sqrt{n}+4)) \\ &= \frac{1}{6}n(\sqrt{n}-1)(\sqrt{n}+4)(\sqrt{n}+1) \\ &= \frac{1}{6}n(n-1)(\sqrt{n}+4) \end{aligned}$$

However, $\sqrt{n}+4 > \sqrt[3]{2n}+2$. Hence, this scheme requires more edges than the optimal choice. For $n = 256$, for example, we find the best choice of a is 32 (and $b = 8$) not $a = b = 16$. This is what our requirement $2a = b^2+1$ would suggest, since $2.32 \approx 8^2+1$. The 2-hop scheme in which $a = 32$ requires 162304 edges, whereas the scheme in which $a = 16$ requires 217600.

5.3 Other Possibilities

Of course, we may decide that we wish to bound the number of derivation steps by some number $h \neq 2$. One obvious reason why we might do this in practice is if the storage costs of the 2-hop scheme are too high. In this case, we might split T_n into copies of T_a and D_a within T_b and define schemes that require h_a and h_b hops, where $h = h_a + h_b$. We can use any of the techniques described elsewhere in this paper to construct schemes that provide the desired trade-off between worst-case derivation time and the amount of public information required. A more systematic exploration of these constructions is beyond the scope of this paper and will be the subject of future work.

A second possibility is that we may require a 3-hop scheme. In this case, providing n has three factors, a, b and c, say, we can split T_n into bc copies of T_a, derive a 1-hop scheme for T_a, and then develop a 2-hop scheme for T_{bc} as above.

6 Concluding Remarks

In this paper we have developed a number of schemes for reducing the derivation time required for keys in cryptographic schemes for implementing temporal access control. This reduction has been achieved at the expense of the addition of a small amount of public information (although the reduction from $m-1$ steps to $\log_2 m$ described in Scheme 2 requires no additional storage). The characteristics of a number of the schemes we have described are summarized in the bottom section of Table 4. In the remainder of this section we consider related and future work.

6.1 Related Work

There are two strands of closely related work. Both strands include schemes in which users may have up to three keys. We focus on schemes in which users have a single key, as they are directly comparable to our schemes.[3] The relevant characteristics of comparable schemes in the literature are summarized in Table 4.

- Atallah *et al* [2] propose a number of schemes that reduce the number of edges and the maximum number of hops for I_m, using techniques they had previously developed for total orders [3].
- Ateniese *et al* [4] and De Santis *et al* [11] propose a number of schemes that take a quite different approach, using existing work due to Thorup [14] and to Dushnik and Miller [13] to reduce the diameter of I_m.

The main distinguishing feature of our work is the focus on improved schemes that are directly relevant to the problem at hand, whereas prior work has simply

Table 4. A comparison of related work

Scheme		Storage	Derivation
Atallah *et al*	[2, §4]	$\mathcal{O}\left(n^2 \log_2 n\right)$	4
	[2, §4]	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}\left(\log_2^* n\right)$
Ateniese *et al*	[4, §4.2]	$\mathcal{O}\left(n^3\right)$	1
De Santis *et al*	[11, §3.1]	$\mathcal{O}\left(n^2\right)$	$\mathcal{O}\left(\log_2 n \log_2^* n\right)$
	[11, §3.1]	$\mathcal{O}\left(n^2 \log_2 n\right)$	$\mathcal{O}\left(\log_2^* n\right)$
	[11, §3.1]	$\mathcal{O}\left(n^2 \log_2 n \log_2(\log_2 n)\right)$	3
Our work	Scheme 1	$\frac{1}{6}n(n-1)(n+4)$	1
	Scheme 2	$n(n-1)$	$\log_2 n$
	Scheme 3 ($b=2$)	$\frac{3}{2}n(n-1)$	$\frac{1}{2}\log_2 n$
	§5.1	$\frac{1}{4}n^2(b+2)$	2

[3] We noted in Sect. 2 that schemes in which users have more than one key are usually considered to be undesirable. Naturally, increasing the number of keys is sure to reduce either the public storage or the key derivation time or both.

applied existing techniques without considering the particular characteristics of the graph I_m and its application to temporal access control. As a consequence, we are able to define tight bounds on storage and derivation costs, whereas related work only describes asymptotic behavior. For large values of n, such a description may be useful, but, for smaller (and arguably more relevant) values of n, our approach is more informative. Moreover, without knowing the "magic" constants hidden by the $\mathcal{O}$ notation, it is difficult to ascertain which scheme in the literature is the best to use for a particular value of n. Finally, we may still be able to apply some of the techniques used by De Santis *et al* to further reduce the storage and derivation time for our schemes.

6.2 Future Work

There are two obvious areas for future work. The first is to investigate how to use the techniques described in Sect. 5 to construct h-hop schemes for a specified value of h. At the moment, we have only considered the case $h = 2$. We also hope to investigate recursive schemes that use the factorization of n to produce h-hop schemes, where h is the number of prime factors in n.

The second strand of research would consider the application of the techniques to lattices of the form $L \times I_m$, where L is a security lattice in the usual information flow sense. These types of lattices – which provide a cryptographic way of enforcing temporal information flow policies – have been considered by Crampton [8], as well as De Santis *et al* [4,11] and by Atallah *et al* [2].

References

1. Akl, S.G., Taylor, P.D.: Cryptographic solution to a problem of access control in a hierarchy. ACM Transactions on Computer Systems 1(3), 239–248 (1983)
2. Atallah, M.J., Blanton, M., Frikken, K.B.: Incorporating temporal capabilities in existing key management schemes. In: Biskup, J., López, J. (eds.) ESORICS 2007. LNCS, vol. 4734, pp. 515–530. Springer, Heidelberg (2007)
3. Atallah, M.J., Frikken, K.B., Blanton, M.: Dynamic and efficient key management for access hierarchies. In: Proceedings of 12th ACM Conference on Computer and Communications Security, pp. 190–202 (2005)
4. Ateniese, G., De Santis, A., Ferrara, A.L., Masucci, B.: Provably-secure time-bound hierarchical key assignment schemes. In: Proceedings of 13th ACM Conference on Computer and Communications Security, pp. 288–297 (2006)
5. Bell, D.E., LaPadula, L.: Secure computer systems: Unified exposition and Multics interpretation. Technical Report MTR-2997, Mitre Corporation, Bedford, Massachusetts (1976)
6. Bertino, E., Bonatti, P.A., Ferrari, E.: TRBAC: A temporal role-based access control model. TISSEC 4(3), 191–223 (2001)
7. Bertino, E., Carminati, B., Ferrari, E.: A temporal key management scheme for secure broadcasting of XML documents. In: Proceedings of the 8th ACM Conference on Computer and Communications Security, pp. 31–40 (2002)
8. Crampton, J.: Cryptographically-enforced hierarchical access control with multiple keys. In: Journal of Logic and Algebraic Programming (to appear, 2009); electronic preprint available from doi:10.1016/j.jlap.2009.04.001

9. Crampton, J., Martin, K., Wild, P.: On key assignment for hierarchical access control. In: Proceedings of 19th Computer Security Foundations Workshop, pp. 98–111 (2006)
10. Davey, B.A., Priestley, H.A.: Introduction to Lattices and Order. Cambridge University Press, Cambridge (1990)
11. De Santis, A., Ferrara, A.L., Masucci, B.: New constructions for provably-secure time-bound hierarchical key assignment schemes. In: Proceedings of 12th ACM Symposium on Access Control Models and Technologies, pp. 133–138 (2007)
12. Denning, D.E.: A lattice model of secure information flow. Communications of the ACM 19(5), 236–243 (1976)
13. Dushnik, B., Miller, E.W.: Partially ordered sets. American Journal of Mathematics 63(3), 600–610 (1941)
14. Thorup, M.: Shortcutting planar digraphs. Technical Report 93-60, DIMACS (1993)

Blunting Differential Attacks on PIN Processing APIs*

Riccardo Focardi[1], Flaminia L. Luccio[1], and Graham Steel[2]

[1] Università Ca' Foscari Venezia
{focardi,luccio}@dsi.unive.it
[2] Laboratoire Spécification et Vérification, CNRS & INRIA & ENS de Cachan
Graham.Steel@inria.fr

Abstract. We propose a countermeasure for a class of known attacks on the PIN processing API used in the ATM (cash machine) network. This API controls access to the tamper-resistant Hardware Security Modules where PIN encryption, decryption and verification takes place. The attacks are differential attacks, whereby an attacker gains information about the plaintext values of encrypted customer PINs by making changes to the non-confidential inputs to a command. Our proposed fix adds an integrity check to the parameters passed to the command. It is novel in that it involves very little change to the existing ATM network infrastructure.

Keywords: Security APIs, Financial Cryptography, PIN Verification.

1 Introduction

In the international ATM (cash machine) network, users' personal identification numbers (PINs) have to be sent encrypted from the PIN entry device (PED) on the terminal to the issuing bank for checking. Issuing banks cannot expect to securely share secret keys with every cash machine, and so the PIN is encrypted under various different keys as it passes through the network. Typically, it will first be encrypted in the PED under a key shared with the server or *switch* to which the ATM is connected. The PIN is then decrypted and re-encrypted under the key for an adjacent switch, to which it is forwarded. Eventually, the PIN reaches a switch adjacent to the issuing bank, by which time it may have been decrypted and re-encrypted several times. The issuing bank has no direct control over what happens in the intermediate switches, so to establish trust, the internationally agreed standards ANSI X9.8 and ISO 9564 stipulate the use of tamper proof cryptographic hardware security modules (HSMs). In the switches, these HSMs protect the PIN encryption keys, while in the issuing banks, they also protect the PIN derivation keys (PDKs) used to derive the customer's PIN from non-secret validation data such as their personal account number (PAN).

* Work partially supported by Miur'07 Project SOFT: "*Security Oriented Formal Techniques*".

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 88–103, 2009.

All encryption, decryption and checking of PINs is carried out inside the HSMs. The HSMs have a carefully designed API providing functions for *translation* (i.e. decryption under one key and encryption under another) and *verification* (i.e. PIN correctness checking). The API has to be designed so that even if an attacker obtains access to the host machine connected to the HSM, he cannot abuse the API to obtain customer PINs.

In the last few years, several attacks have been published on the APIs in use in these systems [7,10,12]. Very few of these attacks directly reveal the PIN. Instead, they involve the attacker calling the API commands repeatedly with slightly different parameter values, and using the results (which may be error codes) to deduce the value of the PIN. High-profile instances of many PINs being stolen from a hacked switch has increased interest in the problem [1]. In April 2009, a Verizon Data Breach report and a subsequent interview with the report's author in the press confirmed publicly for the first time that PINs are being extracted from HSMs on a wide scale, and that organised criminal gangs are actually implementing attacks "that a year ago were thought to be only academically possible" [2,3].

We are concerned with researching improvements to the PIN processing infrastructure that could mitigate these attacks. PIN recovery attacks have been formally analysed before, but previously the approach was to take a particular API configuration and measure its vulnerability to combinations of known attacks [16]. In recent work, we proposed an extension to language based information flow analysis to take account of cryptographic primitives designed to assure data integrity, in particular MACs [11]. We showed how PIN processing APIs could be extended with MACs to counteract differential attacks, and showed how this revised API type-checked under our framework. However, that work was rather theoretical, and did not attempt to explain how our proposal could be put into practice. In particular, for our proposal to be feasible in the short term, it needs to be adapted to take into account the constraints of the existing ATM infrastructure. In this paper we outline what we believe to be a practical scheme. We assess its impact on security and the amount of changes that would be required to the ATM network to put it into practice.

The rest of this report proceeds as follows: in section 2 we first review the PIN processing APIs we are interested in, and then in section 3 the known attacks. In section 4 we detail our fix, show how it improves security, and discuss the implementation of our proposal. In section 5 we compare our scheme to other proposals in the literature, we then conclude in section 6.

2 Background

In this section we first introduce the architectural model and we then revise some of the security APIs used for financial *Personal Identification Number* (PIN) verification. We focus on the ones that have been shown to be critical for PIN security when the HSM is in *operational mode*, i.e. when it is processing normal transactions from ATMs. Most HSMs support a further mode of operation such

as *privileged mode* or *security officer login* which is used, e.g., for loading new cryptographic keys, but generally this mode requires physical access to the HSM in order to enter codes on a keypad or present smartcards. Attacks at this level are considered out of scope for this paper.

2.1 The Architecture

Let us recall the ATM network structure [12]: the customer inserts the card and keys in the related PIN at and ATM of the *acquiring* bank. The PIN has to travel to the *issuing* bank[1], i.e. the bank where the customer has the account. Most of the time the acquiring bank and the issuing bank do not coincide, and may be located at great distance. In this case the PIN has to travel along a network of intermediary switches, encrypted via different symmetric keys: the issuing bank shares a key with the first neighbouring switch, this switch shares another key with the next switch, and so on. Thus, messages arriving at a switch have to be decrypted and re-encrypted with the new key. When messages containing the PIN arrive at the issuing bank, this bank has to confirm the association between the PIN and the *Personal Account Number* (PAN), i.e. an identifying account number associated with each users account.

2.2 PIN Management APIs

In the previous section we have observed how PINs travelling along the network have to be decrypted and re-encrypted under a different key: This is done using a *translation* API. There is a further complication however: PINs have to be formatted into 64-bit blocks for encryption under the 3DES cipher used in the network. Different countries, manufacturers, and banking organisations have proposed different formats for the block, some of which are now standardised in ISO 9564. The different block formats reflect the different capabilities of different nodes in the network (for example, some require the encrypting device to generate a fresh random number). Not all switches are able (or willing) to support all known PIN formats. Thus, the translation function might need to reformat messages under newer or older formats accepted by the next switch. In some cases, e.g., when the output format is ISO-0, the PAN related to the PIN has also to be specified. When the PIN reaches the issuing bank, its correspondence with the PAN is checked via a *verification* API. Even at the verification API, several different PIN block formats might be supported.

ISO-1 and ISO-0 formats. We now illustrate ISO-1 and ISO-0 formats, as we will often refer to them when describing the attacks. We report all the other commonly used formats in appendix A.

[1] Chip based cards now support the possibility of checking the PIN offline, i.e. checking it on the chip. However, under most schemes, this is intended for point-of-sales transactions. For cash advances and ATM transactions, an online PIN check is still required. Furthermore, large countries like the USA do not have chip-based card schemes in widespread use.

ISO-1 supports PINs from 4 to 12 digits in length. The formatted PIN appears as follows:

1	L	P	P	P	P	P/R	P/R	P/R	P/R	P/R	P/R	P/R	P/R	R	R

In particular, 1 identifies the ISO-1 format, L is the length of the PIN, P are the PIN digits while P/R is either a PIN digit or the pad random value R, depending on L. The random padding aims at avoiding codebook attacks.

ISO-0 is equivalent to ANSI X9.8, VISA-1, and ECI-1. This format is not randomized but uses the PAN to prevent codebook attacks. The 4-12 digits PIN is first formatted as follows:

0	L	P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	F	F

0 stands for ISO-0, L is the length of the PIN, P are the PIN digits while P/F is either a PIN digit or the pad value F, depending on L. To obtain ISO-0, first the rightmost 12 digits of the PAN are written as follows:

0	0	0	0	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN

and then the two numbers above are xored:

0	L	P	P	P xor PAN	P xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	F xor PAN	F xor PAN

Translate and Verify API. We now specify more in detail the two APIs.

- PIN_Translate(k_I, k_O, $format_I$, $format_O$, PAN_I, PAN_O, EPB_I, EPB_O)
  This API takes a PIN block EPB_I encrypted with the key referred by k_I and formatted as $format_I$, extracts the PIN, reformats it as $format_O$ and re-encrypts it with key referred by k_O. When one of the two formats is ISO-0, the PAN is also provided via PAN_I or PAN_O.
  When the API succeeds, the obtained PIN block is returned in EPB_O. Otherwise, an error code is returned.
- PIN_Verify(k_I, k_V, $format_I$, PAN_I, EPB_I, v_data)
  This API takes a PIN block EPB_I encrypted with the key referred by k_I and formatted as $format_I$, extracts the PIN and checks its validity via the verification key referred by k_V and the verification data v_data. For example, in the IBM 3624 PIN calculation method [13], v_data is the triple ⟨offset, validation_data, dec_tab⟩: a user PIN of length[2] l is obtained by decimalising through dec_tab the first l digits of the encryption $enc(k_V, \text{validation\_data})$, and by digit-wise summing modulo 10 the offset. Once the user PIN is recovered, its equality is checked against the received encrypted PIN, i.e. the one inserted at the ATM.
  The API returns the result of the verification or an error code.

[2] The PIN length is either encoded in the format or passed as a further parameter, as mentioned below.

The above described APIs are slight abstractions and a simplifications of real ones. For example, referring to [13], we have that PIN_Translate(k_I, k_O, $format_I$, $format_O$, PAN_I, PAN_O, EPB_I, EPB_O) corresponds to Encrypted_PIN_Translate except that we have omitted parameters rule_array_count, rule_array, and sequence_number. The first two are used to specify whether the API should only *translate* or *translate-and-reformat* the encrypted PIN, and in the latter case a PIN extraction method has to be specified. Here, we are assuming the API always performs a *translate-and-reformat*, but we can recover the simpler *translate* case, e.g., by just passing NULL in the $format_O$ parameter. Finally, the parameter sequence_number is not used and always points to a constant.

PIN_Verify(k_I, k_V, $format_I$, PAN_I, EPB_I, v_data) corresponds to Encrypted_PIN_Verify, where we have omitted the parameters rule_array_count, rule_array, described above, and PIN_check_length, which is used to specify the length of the PIN for the format IBM 3624.

3 Known Attacks

In this section, referring to [7,10,12,14], we give a brief overview of known attacks on cryptographic APIs. We will omit some of the details, mainly concentrating on the specification of the API calls involved. All the attacks have the same scenario: an attacker has obtained access to a host machine at a switch or verification facility, and is able to call any function of the API with any parameters. Additionally he is assumed to be able to intercept some genuine traffic arriving for processing at the HSM (i.e. encrypted PIN blocks and associated data from real ATM transactions). However, he is assumed to be *unable* to obtain, by brute force search or other means, the PIN encryption keys or PIN derivation keys in use in the system.

3.1 Attacks on Verification API

We now look at attacks on the verification API, which we split into two categories.

Decimalisation Table Attacks. This family of attacks was discovered by both Bond and Zieliński, [10], and Clulow, [12, §3.5.5]. Since a proposal by IBM (the so-called '3624 scheme'), many PIN schemes assign initial customer PIN values by encrypting a customer's PAN under a secret PIN derivation key (PDK), and then decimalising the result using a decimalisation table (or 'dectab'). A decimalisation table maps each hexadecimal value to a decimal. The 'standard' decimalisation table looks like this:

Hex. value	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Dec. value	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5

So, if the first four digits of the result of encrypting a customer's PAN under the PDK are 4A6B hexadecimal, the assigned PIN will be 4061. Not all banks

use the same decimalisation table [6], so the table is passed as a parameter to the verify command. Some schemes allow the customer to change her PIN at an ATM. This is achieved by fixing an offset, which when added digitwise modulo 10 to the original PIN, gives the customer's chosen PIN. This offset is not considered to be security critical, since without the original PIN it provides no help in guessing the correct customer PIN.

Decimalisation table attacks do not determine the PIN digitwise, but rather determine first what digits are in the PIN, and then where these digits are. Suppose an attacker has an encrypted PIN block which, when supplied along with the standard decimalisation table and a known offset, correctly verifies inside the HSM (that is, the HSM reports that the PIN is correct when the PIN_Verify(.) function is called). Now suppose the attacker alters the decimalisation table like this:

Old value	0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
New value	1 1 2 3 4 5 6 7 8 9 1 1 2 3 4 5

He then calls PIN_Verify(.) again, with the modified dectab. If the verification still passes, then he knows there are no 0s in the PIN. If however the verification now fails, he knows there must be at least one 0 somewhere in the PIN. The problem now is to determine how many there are, and where. This can be accomplished by altering the offset. The attacker advances the offset by one at each position, and then at every combination of positions, until the PIN is once again reported as being correct. This reveals the location of the 0s in the PIN. The table below illustrates the process, for an example where the customer's PIN is 3060, and the offset is 0000. We assume the attacker has already tried the modified dectab shown above, and discovered that there is at least one 0 somewhere in the PIN.

Attacker set offset	Result from HSM	Knowledge of PIN
0001	Incorrect PIN	????
0010	Incorrect PIN	????
0100	Incorrect PIN	????
1000	Incorrect PIN	????
0011	Incorrect PIN	????
0101	Correct PIN	?0?0

The decimalisation table attack takes an average of 16.145 API calls to determine a four-digit PIN [16, §5].

Check Value Attack. Clulow has described an attack on the PIN verification API that takes advantage of the 'check value command' [12, §3.5.8]. Many APIs support such a command, used to ensure that a key has been imported correctly. The function of the command is to return a 64 bit block of 0s encrypted under a given key.

To make the attack, the attacker must also be able to supply a block of 0s as validation data to a command for verifying PINs such as the one described above. The first step is to obtain the check value of the PDK, and decimalise the

first four characters of the result using the standard decimalisation table. Store this as IPIN. Now, supply a PAN of 000000000000 to the function PIN_Verify(.), along with some encrypted PIN block (EPB) you want to crack. Start with offset 0000. Generally, the command will at first fail. Increase the offset by 1 until the command reports a successful verification. Store the final offset as OFFSET. Now we know that the PIN in the block verifies successfully when compared against IPIN + OFFSET (mod 10 each digit), and so this must be the customer's PIN.

On average, for a four-digit PIN, this attack would require one call to the check value command and 5000 calls to the PIN verify function.

3.2 Attacks on the Translate API

We now consider two different attacks on the translate API.

The codebook attack. This attack uses the translation API to get rid of variant information from the EPBs. This makes EPBs containing the same PIN equal, thus enabling standard codebook attacks. Notice that if EPBs contain random padding, as e.g. in the ISO-1 format, or if the PAN is xored to the PIN before it is encrypted, as happens for ISO-0, equal PINs in general produce different EPBs.

The attack can be applied both to switches and to verification facilities [7]. The attacker first generates all the EPBs for every possible PIN of a fixed length, in any ISO format. This can be done, e.g., by a network of attackers trying all of the 10000 PINs at different ATMs and intercepting the corresponding EPB at the hacked switch. A smarter method, described in [7], only requires 100 trials at the ATMs, but we do not describe it here for lack of space. Once the 10000 EPBs have been obtained, the attackers translate, at the switch where they have been intercepted, all these EPBs into ISO-0 format with a chosen fixed PAN, say PANA, and create a look-up table with all the EPBs and the associated PINs. At the arrival of real EPBs, they translate them into an ISO-1 format (removing the PAN) and then reformat them into an ISO-0 message together with the chosen PANA. The real PIN can now be revealed by searching this new encryption in the look-up table.

Wrong-format attack. This attack [12,16] consists of translating a received encrypted message into an ISO-0 format with different PANs, and to acquire information depending on the return value of the translate API. More precisely, the attacker provides different PANs (PANA^i), which are xor-ed to the decrypted value to recover the PIN. When such an xor operation gives non-decimal values, the API returns an error code. This gives information about the actual PIN digits. In fact, for an n digits PIN, by repeating this operation $16(n-2)$ times it is possible to recover the $n-2$ rightmost PIN digits up to their parity. For example, for a 4 digits PIN we might discover that the third and fourth digits are 3 or 4 and 7 or 8, respectively. Notice that the first two digits cannot be recovered because they are not xor-ed with the PAN.

There is a trick to recover also the first two digits of the PIN: the attacker first translates a message in ISO-0 (where the PIN starts from the 3rd position) with

a PAN of all 0s, and then claims it is in a VISA3 message format (that starts with the digits of the PIN ended by a sequence of Fs), and asks to translate it into ISO-0 with a PAN of all 0s. The result of this operation is that the PIN is shifted two positions to the right, thus permitting the attacker to use the previous attack on the entire PIN.

Finally, the choice between the pair of values can be narrowed with 4 more calls: Assume the pairs of values are $\{P_1, P_1 + 1\}$, $\{P_2, P_2 + 1\}$, $\{P_3, P_3 + 1\}$, $\{P_4, P_4 + 1\}$, i.e. $2^4 = 16$ possible PINs, the attacker can now pass a message in an ISO-0 format claiming is a VISA3, together with PAN $E_1 00000000000$, with $E_1 = E \oplus P_1$. If the HSM outputs an error, the right PIN digit value is P_1, else is $P_1 + 1$. In fact, the given PAN is xored to the PIN and if the PIN digit value is P_1 we obtain $P_1 \oplus E_1 = P_1 \oplus E \oplus P_1 = E$ which is non-decimal and gives an error. To recover the other values P_2, P_3 and P_4, the attacker provides PANs $0E_2 0000000000$, $00E_3 000000000$, $000E_4 00000000$, respectively, where $E_i = E \oplus P_i$.

Attacks on other APIs. Attacks have also been found on the functions used for customer PIN change [9] and secure messaging [4]. These are out of scope for the current paper, but we plan to treat them in future work.

4 Our Proposed Fix

To understand the rationale behind our fix, first observe that the attacks explained in section 3 are 'differential attacks' in the sense that they involve an attacker learning about the value of the PIN by tweaking parameters to the API commands. The name comes from the similarity to differential attacks on block ciphers, where certain bits in the inputs to the cipher are changed and the result observed [8]. One way to prevent these attacks would be to prevent 'unauthorised' queries to the API. In theoretical terms, we are trying to achieve 'robust declassification' [15]. We know that the functions of the API must declassify some data, specifically they must tell us whether the encrypted PIN is the correct one or not. Robust declassification means an intruder cannot influence what data is declassified. However, the existing APIs have no way of distinguishing a legitimate query from an illegitimate one.

In another paper [11], we show how in general differential attacks can be countered by the use of MACs. The idea is to add a MAC of the non-confidential parameters required for PIN processing to the input to the relevant API command. The command checks the MAC before performing the PIN operation. If the MAC check fails, the command halts without processing the PIN. This way we achieve 'robust declassification'. However, the infrastructure changes needed to add full MAC calculation to ATMs and switches are seen as prohibitive by the banks [6]. We propose here a way to implement a weaker version of our scheme whilst minimising changes to the existing infrastructure. We lose some security, since our MACs now have an entropy of only 5 decimal digits ($2^{16} < 10^5 < 2^{17}$). We assess the effect of this change in section 4.3.

4.1 CVC/CVV Codes

We observe that cards used in the cash machine network already have an integrity checking value: the card verification code (CVC), or card verification value (CVV), is a 5 (decimal) digit number included on the magnetic stripe of most cards in use. It is, in effect, a MAC of the customer's PAN, expiry date of the card and some other data. The current purpose of the CVV is to make copying cards more difficult, since the CVV is not printed on the card and does not appear on printed POS receipts[3]. Below we give the algorithm for calculating the CVV. This is done at the card issuing facility. CVVs are checked at the verification facility.

PAN	Exp date	Service code	0 pad
16 digits max	4 digits	3 digits	9 digits max
Block B1	Block B2		

Each decimal digit is encoded in 4 bits. Note the partition of the CVV plaintext field into two blocks, B1 and B2. To construct the CVV, a two-part DES key is required. Call the two 64-bit parts key K1 and key K2. The hexadecimal CVV root is constructed as

$$CVV_{hex} := enc(K1, dec(K2, enc(K1, (enc(K1, B1) \oplus B2))))$$

The 5 digit decimal number is constructed using the Visa decimalisation scheme:

1. Extract all decimal digits from CVV_{hex}, preserving their order from left to right.
2. Left justify the result
3. Reduce any remaining digits by 10
4. Left justify the result and append it to the result of 2
5. The CVV is the first 5 digits (from left to right) of the result.

4.2 Packing the MAC into the CVC/CVV

Our proposal is to pack more information into the CVV at issue time, and to use this as a MAC at the verification facility.

In our formal scheme, we included the data of the decimalisation table and card offset. Observe that with the maximum 16 digits of the PAN being used, we still have 9 digits of zeros in the final field of block B2. Our idea is to use the CVV calculation method twice, in the manner of a hashed MAC or HMAC function. We will calculate the CVV of a new set of data, containing the decimalisation table and offset or PVV and a code for the PIN block format. Then we will insert the result of the original CVV calculation to produce a final 5-digit MAC.

[3] The CVV/CVC is not to be confused with the CVC2 or CVV2, which is printed on the back of the card and designed to counteract 'customer not present' fraud when paying over the Internet or by phone.

Our second CVV, which we will call CVV', contains the following fields:

Dectab	Offset/PVV	PIN block format	original CVV	0 pad
16 digits max	4 digits	1 digit	5 digits	6 digits max
Block B1'	Block B2'			

The position of each field is not important (except that it has to be fixed and agreed). We calculate CVV' in the same way as the standard CVV. This makes for easy upgrade from the original infrastructure, because CVV generation and verification commands are already available in HSMs so will need minimal changes to the firmware. The PIN_Verify command of the HSM must be changed to check CVV' before performing a verification test. The PIN test is only performed if the CVV' of the inputs matches the supplied CVV'.

One could use the same keys for calculating the CVV' as for the CVV, or one could use different keys. Either way, the verifying HSM needs access to these keys. The use of different keys could be motivated by a desire to be able to check CVVs, and so to an extent to verify card authenticity, at other points in the ATM network, without giving these nodes the keys used to create CVV's.

4.3 Security of the CVV Based Scheme

In our formal scheme we gave a proof of security by typing [11]. We proved that the modified version of the PIN_Verify function, implementing the MAC check before checking the PIN, gives robust declassification of the correctness of the PIN, subject to the following assumptions: 1) the MAC is unforgeable, 2) the attacker only has one encrypted PIN block to test, containing the correct PIN for a single given PAN, 3) encryption is perfectly secure (though not necessarily randomised). In proposing a scheme with a 5-digit MAC value, we are admitting the possibility of brute-force attacks, and thus breaking assumption 1). However, to guess the CVV' for a given set of parameters should take an average of 50 000 trials. So, an attack like the dectab attack which previously required 15 calls to the API will now take 750 000 calls. HSMs typically perform something of the order of 1000 PIN verifications per second, so this change moves the attack time for a single PIN from 0.015 seconds to 750 seconds, or 12.5 minutes.

Of course, the security of the scheme depends on the attacker not being able to calculate CVV's by any other means that brute force. For example, the API of the HSM must not make available the functionality to allow the creation of CVV's on arbitrary data. However, one would imagine this functionality only being available at an issuing facility (much like the functionality that prints PIN mailers to send to customers).

4.4 Practicalities of Deploying Our Proposal

Sources close to the banking industry have told us the hardest (i.e. most costly) things to change in the ATM network are the network itself and the messaging formats. Our proposal has the advantage of requiring neither of these to change.

ATMs and switches generally already send the CVV from the ATM to the issuing bank, so can easily be adapted to send CVV'. In fact, many ATMs blindly send all the 'Track 2 data' from the magnetic card - this includes the PAN, expiry date, and CVV. Under most schemes there is still space on the magnetic stripe for a further 5 digit code. Chip based cards should have no problem storing a further 20 bits of data. So, we could use CVV' and the original CVV, thus allowing old style CVVs to be checked separately if required.

The next most costly thing to change is the software in the bank that calls the HSM. Our proposal requires some small changes here, since the verification command will now have extra parameters (the CVV' and a handle for the CVV' key). Changing the firmware in the HSM is not considered to be a prohibitive expense, and we require only minor changes, since HSMs can already verify CVVs.

A further advantage of our proposal to put the MAC onto the magnetic stripe is that it does not require any changes to the firmware of the PED. No MACs need to be calculated at the ATM, and no MAC keys need to be distributed to ATMs.

A particular feature of our scheme is that it does not require banks not adopting the scheme to change anything. A bank can unilaterally decide to add CVV's to its cards and require them at its verification facilities. The CVV' will pass through the network along with the rest of the track 2 data. The bank will thereby gain assurance of security at its verification facilities. However, it is known that PINs are not always verified at the issuing bank. There are also 'stand-in' verification centres run by, e.g., Visa and Mastercard, deployed to assure availability or to deal with foreign ATM transactions in certain cases. Upgrading these will be more complex, since even if the new 'Verify with CVV" command is made available at the facility, if an intruder can still access the old command, he can *a priori* still make the old attacks. A facility could work around this by partitioning the verification keys (PDKs) such that PDKs designated for use in the new scheme can only be used under the new command. Features for this kind of *key separation* are already implemented in many HSMs, since it is known to be dangerous to allow, for example, a key for IBM PIN verification to be used for VISA PVV verification [12, §3.5.7].

4.5 Addressing Translation Attacks

So far, our proposal only addresses the problem of attacks at the verification facility. By preventing queries with altered dectabs, PANs, or offsets, we blunt the attacks given in section 3.1. However, the CVV' cannot be used to ensure integrity of parameters at translation nodes, in order to address the attacks of section 3.2. This is because firstly, we do not envisage distributing the CVV' keys to all the nodes in the network, and secondly, we need to add a parameter giving the format of the PIN block to the MAC to protect against 'wrong format' attacks, and this parameter will change as the EPB makes its way through the network, requiring the MAC value to be recalculated. The CVV' is therefore not suitable for this purpose.

Point-to-point MACs. There are other ways to upgrade the network to protect against the translation attacks. For example, we can demand that each switch shares a MAC key with every switch to which it might send an EPB, just as it currently shares a PIN encryption key with every such node. It can then calculate a MAC for every outgoing block, and verify a MAC for every incoming block. This is the scheme we proposed in our theoretical paper [11]. In practice, we cannot assume that all switches in the network will be upgraded at once. As a consequence, there will be a time when there are some switches which can calculate MACs, and some that cannot. As a consequence there will be edges in the network (i.e, some communicating pairs of switches) that can deal with MACs and share a key, along with some that cannot. PINs travelling through non-upgraded switches will, of course, still be vulnerable to translation attacks, even if they come from, or travel to, upgraded nodes. In fact, once they reach non-upgraded switches, i.e. switches whose translation function is not MAC-enabled, all the old attacks will be possible. This is, in a sense, natural as we cannot guarantee the security of PINs passing through insecure subnetworks.

Note that even if a PIN does not travel through non-upgraded switches, it might reach a MAC-enabled switch that is in contact with non-upgraded ones (see figure 1). This 'borderline' switch will have to offer both MAC-enabled and standard translate functions in order to communicate with the old protocol. This leaves the door open for attacks as an intruder might just disregard the MAC and call the old translate function in order to mount all the old translate attacks. A switch, in fact, only becomes secure once the old translate command can be disabled. A way to mitigate this problem is to add new special (temporary) *borderline switch* (see figure 2) with the only aim of interfacing upgraded and non-upgraded subnets. This nodes would only be reached by EPBs going to/coming from an insecure, non-upgraded, subnetwork. In this way EPBs travelling inside an upgraded network are not exposed to attacks on borderline nodes.[4]

Single format. Since translation attacks are based on *reformatting* the EPB, another possibility to prevent them would be to enforce the use of one single block format. In this way a translate would only consist of decrypting and re-encrypting the PIN block. Even in this case, upgrading the whole network would require time, leading to the same situation discussed above.

This solution does not require any new modified HSMs and protocols, since it only aims at reducing the existing functionalities so to operate on a single format thus it is, in principle, very appealing. However there are some difficulties that have to be carefully considered: (*i*) the 'political' decision on which format should be elected as 'the one' is far from being easy to take since many different standards exist, proposed by different entities; (*ii*) the choice of the format might require upgrading some old HSMs or PEDs so to support it, especially for randomized formats like ISO1 or ISO3 which require good random number generators not present in old hardware.

[4] There might be attacks on the PIN routing that force them to reach untrusted part of the networks. They are of course relevant but out of the scope of this work.

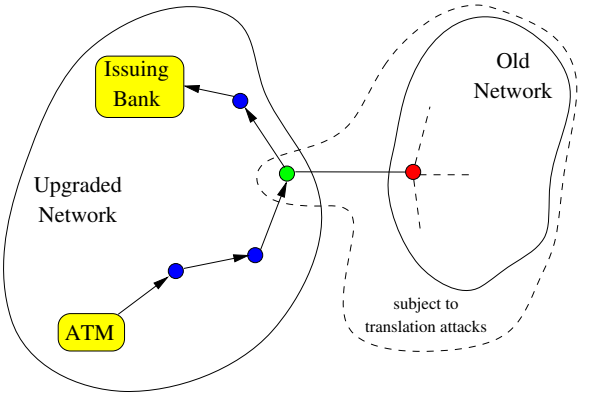


Fig. 1. Problem with borderline switches

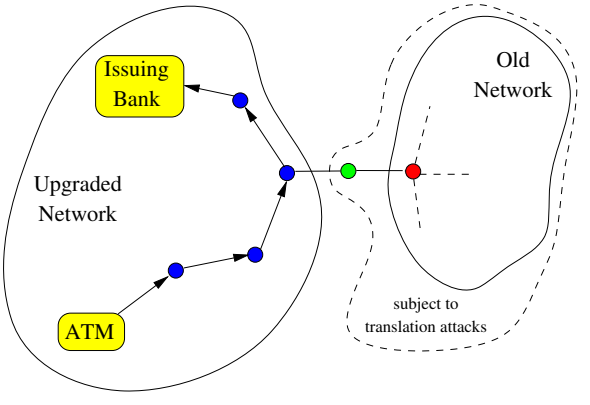


Fig. 2. Adding special borderline switches makes the upgraded network secure

5 Comparison to Other Proposals

Attacks on PIN processing APIs have been known for years [5,7,10,12,16], yet as other authors have observed, "proposals to counter such attacks are almost non-existent in the literature" [14]. In their decimalisation table attack paper, Bond and Zielinski suggest that the "decimalisation table input must be cryptographically protected so that only authorised tables can be used" [10, §6], though they do not give a scheme for doing so. Our proposal is in essence just such a scheme, though it additionally protects the other non-confidential inputs to the PIN verification command. Clulow includes the issue of 'Parameter integrity' in his discussion of solutions to the problem of PIN recovery attacks [12, §3.8]. He writes, "perhaps the most logical solution is to include the use of MACs (or similar methods) to ensure data integrity. For example, a decimalization table is supplied to a function call along with a MAC and the key for verifying that MAC." Our scheme achieves the same objective with a single MAC for all the non-confidential

parameters. Clulow notes that "These solutions are fairly obvious. The challenge comes in ensuring interoperability between devices from different manufacturers". By basing our MAC around an existing standardised algorithm already in use in HSMs, and by putting the MAC onto the card rather than requiring it be calculated elsewhere in the network, we hope to address this challenge.

Finally, Mannan and van Oorschot have presented a scheme for 'salting' PINs [14]. The idea is to have banks fix a 128-bit salt value for each customer. This is written in plaintext to the bank card and stored encrypted at the verification facility. When a customer types her PIN at an ATM, the ATM generates a 12 (decimal) digit number as a pseudorandom function of the typed PIN and the salt. This is called the Transport Final PIN (TFP). This is sent in the EPB to the verification facility, where the process can be duplicated using the customer's true PIN, offset and encrypted salt and a comparison made to see if the PIN was correct. The main advantage of this is that at translation nodes, the known attacks reveal only the salted PIN, not the real one. Given the entropy of the salt, this is of little use to an attacker.

The Mannan/van Oorschot scheme shares some of the advantages of our proposal in that it does not require a change to messaging formats in the network, nor does it require changes to HSMs used in switches. However, their scheme does require that ATM PEDs are changed to be able to calculate a particular pseudo-random function, and also that verification HSMs have access to a database of all encrypted salts. Their proposals are primarily aimed at blunting the attacks of Berkmann and Ostrovsky [7], and do not seem to prevent, e.g., decimalisation table attacks at the verification facility. One could in fact deploy their scheme in conjunction with ours to provide effective protection against a larger set of attacks.

6 Conclusions

We have described a version of a MAC based scheme for ensuring integrity of queries to PIN processing APIs that is easy to implement and does not require wholesale changes to the ATM infrastructure. This ease is at the cost of some security, since the CVV codes can be cracked by brute force, but its implementation in the short term would make attack scenarios far less profitable. In the medium term we feel that the full MAC scheme should be used. The cost of this should be weighed against the cost of a complete overhaul of the way PIN processing is carried out in the ATM network.

References

1. Hackers crack cash machine PIN codes to steal millions. The Times online, http://www.timesonline.co.uk/tol/money/consumer_affairs/article4259009.ece
2. PIN Crackers Nab Holy Grail of Bank Card Security. Wired Magazine Blog Threat Level, http://blog.wired.com/27bstroke6/2009/04/pins.html

3. Verizon Data Breach Investigations Report (2009), http://www.verizonbusiness.com/resources/security/reports/2009_databreach_rp.pdf
4. Adida, B., Bond, M., Clulow, J., Lin, A., Anderson, R., Rivest, R.L.: On the security of the EMV secure messaging API (2008)
5. Anderson, R.: Why cryptosystems fail. Commun. ACM 37(11), 32–40 (1994)
6. Anderson, R.: What we can learn from API security. In: Christianson, B., Crispo, B., Malcolm, J.A., Roe, M. (eds.) Security Protocols 2003. LNCS, vol. 3364, pp. 288–300. Springer, Heidelberg (2005)
7. Berkman, O., Ostrovsky, O.M.: The unbearable lightness of PIN cracking. In: Dietrich, S., Dhamija, R. (eds.) FC 2007 and USEC 2007. LNCS, vol. 4886, pp. 224–238. Springer, Heidelberg (2007)
8. Biham, E., Shamir, A.: Differential cryptanalysis of DES-like cryptosystems. J. Cryptology 4(1), 3–72 (1991)
9. Bond, M.: Understanding Security APIs. PhD thesis, University of Cambridge, England (2004), http://www.cl.cam.ac.uk/~mkb23/research.html
10. Bond, M., Zielinski, P.: Decimalization table attacks for pin cracking. Technical Report UCAM-CL-TR-560, University of Cambridge, Computer Laboratory (2003), http://www.cl.cam.ac.uk/TechReports/UCAM-CL-TR-560.pdf
11. Centenaro, M., Focardi, R., Luccio, F.L., Steel, G.: Type-based Analysis of PIN Processing APIs. In: 14th European Symposium on Research in Computer Security, ESORICS 2009. LNCS (to appear, 2009)
12. Clulow, J.: The design and analysis of cryptographic APIs for security devices. Master's thesis, University of Natal, Durban (2003)
13. IBM Inc. CCA Basic Services Reference and Guide for the IBM 4758 PCI and IBM 4764 PCI-X Cryptographic Coprocessors. Technical report. Releases 2.53–3.27 (2006), http://www-03.ibm.com/security/cryptocards/pcicc/library.shtml
14. Mannan, M., van Oorschot, P.: Reducing threats from flawed security APIs: The banking PIN case. Computers & Security 28(6), 410–420 (2009)
15. Myers, A.C., Sabelfeld, A., Zdancewic, S.: Enforcing robust declassification and qualified robustness. Journal of Computer Security 14(2), 157–196 (2006)
16. Steel, G.: Formal Analysis of PIN Block Attacks. Theoretical Computer Science 367(1-2), 257–270 (2006)

A Some Standard 64-Bit PIN Block Formats

IBM 3624 This format supports PINs from 1 to 16 digits in length. P are the PIN digits. Depending on the PIN length, P/X can be a PIN digit or a pad value, i.e. a value different from all the PIN digits and fixed a-priori.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
P	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X	P/X

ISO-0, ANSI X9.8, VISA-1, ECI-1 Supports PINs from 4 to 12 digits in length.

0	L	P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	F	F

0 identifies the format, L is the length of the PIN, P are the PIN digits while P/F is either e PIN digit or the pad value F, depending on the PIN length. Then the rightmost 12 digits of the PAN are written as follows:

0	0	0	0	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN

and we now just xor the two numbers:

0	L	P	P	P xor PAN	P xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	P/F xor PAN	F xor PAN	F xor PAN

ISO-1 Supports PINs from 4 to 12 digits in length.

1	L	P	P	P	P	P/R	P/R	P/R	P/R	P/R	P/R	P/R	P/R	R	R

1 identifies the format, L is the length of the PIN, P are the PIN digits while P/R is either e PIN digit or the pad random value R, depending on the PIN length.

ISO-2 Supports PINs from 4 to 12 digits in length.

2	L	P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	F	F

2 identifies the format, L is the length of the PIN, P are the PIN digits while P/F is either e PIN digit or the pad value F, depending on the PIN length.

ISO-3 Supports PINs from 4 to 12 digits in length. It is the same as ISO-0 except that the PIN is padded with random digits.

3	L	P	P	P	P	P/R	P/R	P/R	P/R	P/R	P/R	P/R	P/R	R	R

3 identifies the format, L is the length of the PIN, P are the PIN digits while P/R is either e PIN digit or the pad random value R, depending on the PIN length. As for ISO-0, the rightmost 12 digits of the PAN are written as follows:

0	0	0	0	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN	PAN

and we now just xor the two numbers:

3	L	P	P	P xor PAN	P xor PAN	P/R xor PAN	P/R xor PAN	P/R xor PAN	P/R xor PAN	P/R xor PAN	P/R xor PAN	P/R xor PAN	P/R xor PAN	R xor PAN	R xor PAN

VISA-3 Supports PINs from 4 to 16 digits in length. P are the PIN digits while P/F is either e PIN digit or the pad value F, depending on the PIN length.

P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F

Characterising Anomalous Events Using Change - Point Correlation on Unsolicited Network Traffic

Ejaz Ahmed, Andrew Clark, and George Mohay

Information Security Institute,
Queensland University of Technology, Brisbane, Australia
{e.ahmed,a.clark,g.mohay}@qut.edu.au

Abstract. Monitoring unused (or *o*n dark *d*efault) IP addresses offers opportunities to extract useful information about both on-going and new attack patterns. In recent years, different techniques have been used to analyze such traffic including sequential analysis where a change in traffic behavior, for example change in mean, is used as an indication of malicious activity. Change points themselves say little about detected change; further data processing is necessary for the extraction of useful information and to identify the exact cause of the detected change which is limited due to the size and nature of observed traffic. In this paper, we address the problem of analyzing a large volume of such traffic by correlating change points identified in different traffic parameters. The significance of the proposed technique is two-fold. Firstly, automatic extraction of information related to change points by correlating change points detected across multiple traffic parameters. Secondly, validation of the detected change point by the simultaneous presence of another change point in a different parameter. Using a real network trace collected from unused IP addresses, we demonstrate that the proposed technique enables us to not only validate the change point but also extract useful information about the causes of change points.

1 Introduction

Different techniques [1,2,3] have been proposed to monitor network traffic for malicious content. One compelling technique is to monitor unused IP addresses spaces [3,4,5,6] as this traffic has no reason to exist. Due to the absence of any legitimate activity associated with such addresses, the traffic observed is a result of different abnormal activities like traffic from hosts infected by worms, traffic generated by network probing tools or viruses, traffic from misconfigured nodes and backscatter traffic from distributed denial of service attacks. This provides an added advantage to network security researchers as the analysis is not complicated or distracted by complex, hard to analyze, legitimate traffic. While monitoring such addresses has been used for forensics [7,8,9,10,11,12] and attack detection [13,14,15], the analysis of traffic is largely a manual process. In this paper, we address this limitation of manual root cause identification and

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 104–119, 2009.

validation by correlating change points among different network traffic parameters. We do this by monitoring multiple network traffic parameters in parallel and applying change detection techniques on each of the monitored parameter separately.

Several characteristics can be used to define normal network behavior including traffic dynamics such as the type of traffic, volume of traffic and delay experienced by the network. One of the key observations in differentiating between normal and anomalous activity is the change in traffic dynamics. During normal operations traffic dynamics usually remain constant or vary slowly over time whereas during malicious activity they no longer remain relatively constant [16,17]. Detection of malicious activities can thus be considered a change detection problem, with the aim being to detect a change in traffic parameters as quickly as possible [16]. In terms of darknet traffic, abrupt changes are likely to be indicative of an outbreak of a new type of malicious activity. The timely detection and automated characterization of such outbreaks is the goal of the research presented in this paper.

In change detection, there is a sequence of observations whose statistical properties change at some unknown point in time and the goal is to detect this change as soon as possible. Usually extraction of information related to the change point is done by manual analysis which is not only time consuming but also limited due to the nature and size of the collected data.

In this paper, correlation of change points among different traffic parameters is used to automatically extract the information related to the anomaly. The significance of the proposed technique is two-fold. Firstly, automatic extraction of information related to change points by correlating change points detected across multiple traffic parameters; and secondly, validation of detected change point by the simultaneous presence of another change point in a different parameter.

The paper is organized as follows. Section 2 provides an overview of the related work and details our contributions. A brief overview of the change detection technique is provided in Section 3. Section 4 provides detail of the proposed change point correlation technique. Section 5 details the data used in this paper and explains the traffic parameter being considered. Experimental results are provided in Section 6. Finally a conclusion and future directions are given in Section 7.

2 Related Work

The idea to use change detection techniques for detecting different anomalous behavior has existed for some time. Change detection has been used in analyzing network traffic to identify different traffic anomalies including worm detection [18,19], denial of service and distributed denial of service attack detection [20,21,22,23], scan detection [17] and anomaly/fault detection [24,25,26]. Due to its simplicity and effectiveness, change detection has also been used in the analysis of unwanted traffic collected from unused IP addresses. In this regard Bu et al. [27] proposed detecting a worm outbreak based on a change in the

inter-arrival time of packets sent by scanners. The authors have used a CUSUM change detection algorithm on packet inter arrival times as the first step in detecting a worm outbreak. The alarm from CUSUM algorithm is then analyzed in a second stage where a maximum likelihood estimation (MLE) is used to confirm the epidemic.

Limthong et al. [28] used the discrete wavelet transformation (DWT) technique to identify anomalies in unused IP addresses. The authors have analyzed three different types of packets namely TCP SYN, TCP SYN/ACK and UDP packets. The focus of the paper was to identify a suitable measurement interval and to analyze different DWT levels. On the other hand no information about detected anomalies is provided.

Ahmed et al. [14] used a nonparametric CUSUM change detection technique to identify unusual behavior in darknet traffic. The authors proposed a sliding window based memory management technique to identify changes in traffic behavior. The validity of the proposed technique had been tested using both a synthetic data set and real network traces collected from an unused IP address block. In this paper the authors have only validated their proposed technique using UDP traces collected from monitored data. In another study [15], the authors have also proposed detection of "nested" anomalous activities i.e. the commencement of a new session of anomalous activity whilst another anomalous activity is occurring .

Although change detection has received considerable attention in recent years, correlation between different change points has not been addressed. This might be because most of the research in this area makes use of a single parameter. However, multivariate and multichannel [29,30] change detection techniques have been proposed in the literature, where multiple traffic parameters are considered for detecting a change, the basic objective is to detect a change in any traffic parameter to raise an alarm.

Ide et al. [31] have used change point correlation to compare multiple time series from dynamic systems. The authors have used a singular spectrum transformation (SST) technique to identify change points in the time series. The similarity between different time series is obtained by visual inspection of change point scores. The focus of our method proposed in this paper is different as it correlates the change points identified in different traffic parameter time series to validate and automatically extract useful information related to change points identified in primary traffic parameter time series.

Use of change point correlation to identify problems in enterprise middle ware systems was proposed by Agarwal et al. [32]. The authors have used change point correlations (simultaneous occurrence of events in different time series in a given window) to identify problems associated with middle ware architectures. The authors have used the difference of means to identify changes in system time series. In order to reduce false alarms the authors have used a pre-selected threshold to identify a change along with pre-defined signatures for possible problems associated with the system. For the analysis of unsolicited traffic such as that observed while monitoring large unused address spaces, the correct formulation of signatures for

various attacks is not possible. The lack of attack signatures, such as for the zero day attacks, limits the use of the proposed technique in analyzing such traffic. In addition, as mentioned by the authors, the effectiveness of the proposed technique largely depends upon the window size over which the difference of mean is calculated. We argue that the use of a fixed size window has two serious drawbacks, first it introduces a delay in detection and secondly the effect of a huge change in parameter behavior will effect the detection of subsequent behavioral changes at least for the duration of the window size. This makes it unsuitable for analysis of malicious network traffic and was addressed by Ahmed et al. [14].

The work most closely related to our work is of Kaplan et al. [33]. The authors have used change point correlation to identify simultaneous activities from multiple brain areas. The basic idea is to treat two change points in different parameters as correlated if thy appear close in time. Although our work is inspired by this work, there is a significant difference between the two approaches both in the change detection technique and data used. Firstly we used the change detection technique presented in [15]. Secondly our objective is to automatically extract information related to change points identified in traffic collected from unused IP addresses, whereas Kaplan et al. [33] have applied this technique on EEG data.

The focus of the technique proposed in this paper is to automatically extract information related to a change point identified in traffic targeting unused IP addresses. In our framework, information extraction is realized by a two stage process: change point detection; and change point correlation. In the first stage the change detection technique described in [15] is applied to the different time series collected from the unused IP address block. In the second stage the correlation among different change points is carried out to automatically extract the information related to the detected change point.

The novelty of the proposed technique lies in its ability to automatically extract information related to change points by correlating change points across different traffic parameters. In addition the proposed correlation based technique can also be used to validate change points. The effectiveness of the proposed technique is validated using 20 months of network traffic collected from a dedicated block of unused IP addresses.

3 Change Detection Technique

For change detection, a nonparametric CUSUM technique provided in [15] has been used. In this section a brief overview of the change detection technique will be provided, for a detailed explanation the interested reader is referred to [15].

Let $N_1, N_2, ..., N_n$ be the sequence of observations from the monitoring system at fixed time instances $t_1, t_2, ..., t_n$. A malicious activity at time t_k will result in the change in statistical properties of the observed traffic parameter. Let μ_k be the mean traffic rate before and $\bar{X_k}$ be the mean traffic rate after a new anomalous activity, then the CUSUM score, S_k, can be calculated using Equation 1,

$$S_k = \max\left\{0, S_{k-1} + N_k - \mu_k - \alpha.\bar{X_k}\right\} \tag{1}$$

where α is a tuning parameter belonging to the interval (0,1) and is considered to be an upper bound on the estimated post-change traffic rate $\bar{X}_k$.

The choice of the tuning parameter, α, affects the performance of the algorithm as selecting too small or too large an α value will increase the false alarm rate. The α value can either be replaced with a constant value based on experimentation or can be calculated dynamically using Equation 2,

$$0 \leq \alpha \leq 1 - \frac{\mu_k + h_k/T}{\bar{X}_k} \tag{2}$$

where h_k is the detection threshold at time t_k and T is the maximum time in which the change should be detected.

The average number of packets at time t_k can be estimated iteratively using an exponential weighted moving average (EWMA) given in Equation 3,

$$\bar{X}_k = (1 - \beta).\bar{X}_{k-1} + \beta.N_k \tag{3}$$

where $0 < \beta < 1$ is a smoothing factor which gives more weight to the current observation.

The CUSUM score, S_k, given in Equation 1 is then subject to a threshold test h in order to detect a change. An S_k value greater than or equal to the threshold indicates a change in parameter properties. The threshold value h at time t_k is given in Equation 4:

$$h_k^{start} = \sigma_{k-1},\ h_k^{end} = 0.25 \times h_k^{start} \tag{4}$$

where h_k^{start} and h_k^{end} are the threshold values for the start and the end of the change point at time t_k and σ_{k-1} is the standard deviation of the data elements in current window at time t_k. In addition, to reduce the false alarm rate, an additional counter τ is used along with h_k^{end} to mark the end of a change point. The alarm is not canceled until timer τ reaches a specified value (see Equation 5).

$$Alarm = if(S_k \leq h_k^{end} \text{ AND } \tau \geq 2, terminate, resume) \tag{5}$$

4 Change Point Correlation

In analyzing traffic, including that collected by monitoring unused IP addresses , the volume and nature of the observed traffic makes manual analysis of the traffic related to a change point a difficult and time consuming task. This is mainly because such traffic is usually unlabeled and largely malicious which makes every single change point worthy of further investigation.

To overcome this limitation, information extraction related to a change point can be specified as a problem of identifying the simultaneous occurrence of two or more change points in different traffic parameters. The change points in multiple parameters are considered to be correlated if the time difference between them does not exceed a time threshold. The simultaneous occurrence of these change

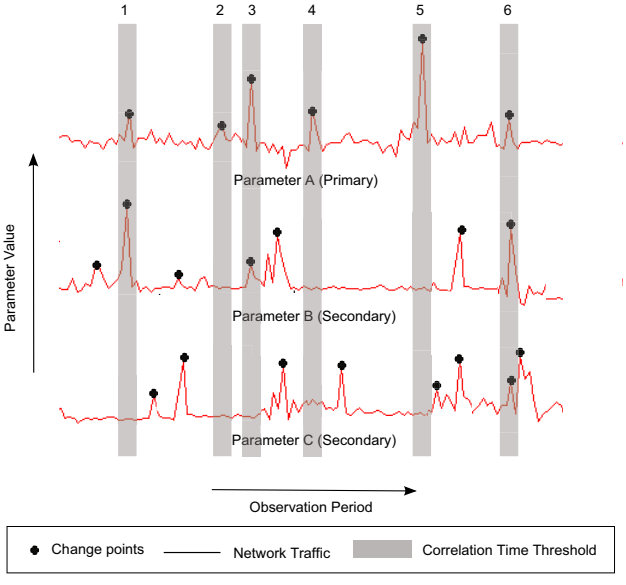


Fig. 1. Correlating change points in different traffic parameters

points in a given time window can then be used to automate the extraction of information related to a change point.

Change points in different traffic parameters are considered to be correlated only if they appear simultaneously within a certain time threshold. This is illustrated in Figure 1 which shows three parameters namely A, B and C. The horizontal axis is the observation period and vertical axis is the measured parameter value. For information extraction, the total number of traffic parameters are divided in two categories namely primary and secondary parameters. For analysis only one primary parameter is selected whereas there can be any number of secondary parameters. The choice of the primary and the secondary parameter is arbitrary and largely depends upon type of network traffic and required analysis. For example the number of packets can be a good candidate for a primary parameter and number of unique sources and number of unique destination ports can both be secondary parameters. In Figure 1, parameter A is a primary parameter while parameters B and C are the secondary parameters.

For change point correlation, each detected change point in the primary parameter is surrounded by a correlation time threshold window, represented by the vertical strips in Figure 1. The change points in the secondary parameters are considered to be correlated with the primary parameter if they fall within the time threshold window. In the above figure the first change point in the primary parameter A is correlated with the change point identified in secondary parameter B, whereas no change point within the time threshold window was identified in secondary parameter C. The close proximity of these change points can be used to explain and character ize the change identified in primary parameter A.

As a single change point can cover more than one time interval, the time threshold window can not be fixed and is calculated as a function of change point duration. Let t_s and t_e be the start and end of a change point then the correlation time threshold window for this change point can be calculated as

$$t_{tr}^{start} = t_s - n \quad , \quad t_{tr}^{end} = t_e + n \tag{6}$$

where t_{tr}^{start} and t_{tr}^{end} is the start and end of correlation time threshold window respectively and n is the correlation threshold allowance having value between 0 and m. The above equation will take n measurement points to the immediate left and n measurement points to the immediate right of the change point identified in primary parameter for correlation analysis.

4.1 Correlation Logic

One of the first questions that arises when correlating change points identified in different network traffic parameters is how to deal with change points identified in different secondary network traffic parameters, with a different change start and end time when compared with t_{tr}^{start} and t_{tr}^{end}. Let $\bar{t}_s$ and $\bar{t}_e$ be the time instances at which the change starts and ends in the secondary parameter. Then for the cases given in Equation 7, the change in the secondary parameter will not cover the correlation time threshold window.

$$(\bar{t_e} \geq t_{tr}^{end} > \bar{t_s} > t_{tr}^{start}) \parallel (t_{tr}^{end} > \bar{t}_e > \bar{t_s} \geq t_{tr}^{start}) \parallel (t_{tr}^{end} > \bar{t_e} > t_{tr}^{start} \geq \bar{t_s}) \tag{7}$$

Keeping in mind the above situations, the correlation logic needs to be enhanced in several ways when considering the duration and the significance of a change point in a secondary parameter with respect to a change in the primary parameter. For information extraction and validation of a change point in the primary parameter it is expected that a change in a secondary parameter will be consistent with the change in the primary parameter. This is given in Equation 8.

$$(\bar{t_e} \geq t_{tr}^{end} > t_{tr}^{start} \geq \bar{t_s}) \tag{8}$$

A simultaneous occurrence of change in both primary and secondary parameters, according to Equation 8, can be used as the key to evaluate these changes. In addition, the significance of the change identified in a secondary parameter can be considered as the magnitude of the change, in the primary parameter, explained by the secondary parameter. This requires that the corresponding change in the primary parameter exceed a certain threshold to be considered significant and can be specified as:

$$(\text{Secondary} \quad \text{Parameter})/(\text{Primary} \quad \text{Parameter}) > X \tag{9}$$

where X is the threshold of the proportion of the change specifying how much of the change in the primary parameter can be explained by the change in a

specific secondary parameter. Since we aim to use correlation for information extraction and validation of a change in the primary parameter, the choice of X will have a significant effect on the correlation. Selecting a large X value means only the secondary parameter which have a significant influence on the primary parameter will be considered in correlation. On the other hand, selecting a small X value will consider secondary parameters with little or no effect on the primary parameter as being correlated, which might not be useful in identifying the cause of a change in the primary parameter.

Although the duration and significance of the change in a secondary parameter gives higher confidence in the extracted information related to the change in primary parameter, the changes in secondary parameters with cases in Equation 7 cannot be ignored. In such cases the change in the secondary parameter is considered correlated with the change in the primary parameter only if the change in the secondary parameter is significant.

Different values for X can be used to correlate changes in secondary and primary parameters. A higher X value for changes in the secondary parameter with a duration according to Equation 7, and lower value if the duration is consistent with a change in the primary parameter (see Equation 8) can be used for correlating changes identified in the primary and secondary parameters. We use $X = 0.3$ where the change in the secondary parameter continues at least for the duration of the change in the primary parameter and $X = 0.5$ where the change in the secondary parameter is within the correlation time threshold window but does not continue for the duration of the change in the primary parameter.

4.2 Change Point Validation

The use of correlation among simultaneous changes in primary and secondary network traffic parameters includes a sort of additional validation of a non-nested (single) and nested change point identified in the primary parameter. These are taken into account only if at least one of the secondary parameters changes simultaneously within the given time threshold window.

In Figure1 for change points 4 and 5, no change in the secondary parameter was identified within the time threshold window. Therefore the changes 4 and 5 in the primary parameter A cannot be validated and hence can be treated as false alarms. Validation of nested change points identified in the primary parameter is confirmed either by the presence of a nested change, or identification of a new change in at least one of the secondary parameters. This changed behavior in secondary parameters is used to confirm the identifica tion of a nested change point in the primary parameter.

The occurrence of false change points, in the present context, does not effect the analysis as the traffic is by definition unsolicited and is either opportunistic or malicious. Every single packet appearing on the unused IP address blocks is unsolicited and we aim to understand what caused it. The simplified algorithm of change point correlation is given in Figure 2.

Algorithm: Change Point Correlation

```
1. for each parameter do
2.       compute change points
3. end for
4. for each change point in primary parameter do
5.       calculate t_tr^start and t_tr^end of the change point
6.       correlation_flag = 0
7.            for each secondary parameter (i) do
8.                 if t_e^i ≥ t_tr^end and t_s^i ≤ t_tr^start then
9.                      if (secondary parameter)/(primary parameter) > 0.3
10.                          change point correlated, correlation_flag = 1
11.                     end if
12.                else if (t_tr^end ≥ t_s^i > t_tr^start) or (t_tr^end > t_e^i ≥ t_tr^start)
13.                     if (secondary parameter)/(primary parameter) > 0.5
14.                          change point correlated, correlation_flag = 1
15.                     end if
16.                end if
17.           end for
18. end for
19. if correlation_flag == 1
20.       change point validated
21. end if
```

Fig. 2. Change point correlation

5 Experimentation

To evaluate the effectiveness of the proposed method, we performed experiments on a real data set collected from a dedicated block of unused IP addresses. The data was collected for a period of about 20 months between 27 November 2006 and 10th August 2008. The monitoring system consists of a single class C address block. For experimentation, the size of the sliding window is set to 100 [14,15]. For correlation, the correlation time threshold allowance (n) is set to 1. Other parameters such as change detection threshold (h_k^{start}), change end threshold (h_k^{end}) and upper bound on mean (α) will be calculated dynamically.

In this experimentation we first categorized the packets based on their type into either TCP, UDP or ICMP. Then for each category different traffic parameters, that have the potential to explain the detected change in the primary parameter, have been extracted. Table 1 provides a summary of these parameters. Even though there are dependencies among these parameters, this does not effect our analysis as we aim to use these parameters in order to understand what causes a change in traffic behavior. Each of these parameters are measured at a regular time interval. First change points were identified in the individual traffic parameter time series using the technique described in Section 3. Change

Table 1. Network traffic parameters

Parameter	Type	Description
N_{pt}^{k}	Primary	Total number of packets of type pt received during k^{th}time interval
S_{pt}^{k}	Secondary	Total number of unique source IP addresses sending packets of type pt during k^{th}time interval
P_{pt}^{k}	Secondary	Total number of unique destination ports receiving packets of type pt during k^{th}time interval
Y_{pt}^{k}	Secondary	Total number of unique payloads in packets of type pt during k^{th}time interval
s_{pt}^{k}	Secondary	Total number of packets of type pt from busiest source address during k^{th}time interval
d_{pt}^{k}	Secondary	Total number of packets of type pt received by busiest destination address during k^{th}time interval
p_{pt}^{k}	Secondary	Total number of packets of type pt received by busiest destination port during k^{th}time interval
$P_{pt}^{k,w}$	Secondary	Total number of packets of type pt received by well known destination ports, (0 to 1023), during k^{th}time interval
$P_{pt}^{k,r}$	Secondary	Total number of packets of type pt received by registered destination ports, (1024 to 49151), during k^{th}time interval
$P_{pt}^{k,d}$	Secondary	Total number of packets of type pt received by dynamic/private destination ports, (49152 to 65353), during k^{th}time interval

point correlation was then used to validate and automatically extract the causes of the change point in the primary parameter using the technique discussed in Section 4. We now provide a discussion of the results.

6 Experimental Results

To evaluate the proposed approach, we have tested it on 20 months of real data collected from a dedicated unused IP address block, class C. The monitored architecture is a passive monitoring, contains no active component and no response is generated, of 256 unused IP addresses. Although the proposed technique has been tested on TCP, UDP and ICMP traffic, we will focus our discussion on UDP analysis. In the absence of any active component, due to the passive nature of the monitoring system, UDP being connectionless protocol is well suited for such an analysis.

Table 2 summarizes the change points identified in the measured traffic parameters for 20 months of the collected traffic. The total number of change points detected in each parameter is given in the total column, this includes both nested and non-nested change points. The nested column provides the number of nested change points identified in each parameter. The last column lists the number of correlated change points. As described in Section 4 for the primary parameter, the total number of UDP packets per unit of time, the correlated column lists the change points for which simultaneous change points were identified in at

Table 2. Summary of the change points identified in UDP traffic

Parameter	Number of Change Points		
	Total	Nested	Correlated
N^k_{udp}	45	12	45
S^k_{udp}	22	5	14
P^k_{udp}	31	10	15
Y^k_{udp}	23	4	10
s^k_{udp}	31	6	19
d^k_{udp}	47	13	13
p^k_{udp}	58	6	38
$P^{k,w}_{udp}$	62	5	19
$P^{k,r}_{udp}$	59	13	21
$P^{k,d}_{udp}$	48	7	6

least one of the secondary parameters. For secondary parameters the correlated column gives the number of change points occurring simultaneously with the primary parameter.

The change points identified in the primary parameter are shown in Figure 3. In this figure the top graph shows the number of UDP packets, primary parameter, observed on the Darknet and the bottom graph shows the change points identified. The vertical axis in the bottom graph represents the outcome of the change detection algorithm with values greater than 1 representing the number of nested changes detected by the change detection algorithm presented in Section 3.

Figure 4 shows the result of correlation of change points identified in primary and some of the secondary parameters for the period of about 9 weeks from 27 November 2006 to 31 February 2007. In this graph only three secondary parameters are shown due to the space limitations, but note that the analysis is performed on all 9 secondary parameters for the period of about 20 months. In this graph the horizontal axis represents time in days, left vertical axis represents number of UDP packets and right vertical axis represents the outcome of change detection algorithm (dotted lines). The correlation time window is shown by the vertical gray portions.

In Figure 4, the first change point in primary parameter is correlated with change point in p^k_{udp}, where k is 30/11/2006. It is also correlated with $P^{k,w}_{udp}$, not shown in the above figure. This helps in not only automatically extracting useful information related to change point but also validates the change point identified in the primary parameter. The observed change point was due to an increase in the traffic on destination port 137, more than 75 port. This port is used by NETBIOS name service and also by Trojan Msinit. The source ports used by these sources were either 137 or 1024+n which confirms the existence of worm looking for unprotected network shares.

The second change point was observed on 7/12/2006 which continued for two days. This change point was correlated with two secondary parameters d^k_{udp} and $P^{k,r}_{udp}$, not shown in the above figure. According to the correlation algorithm

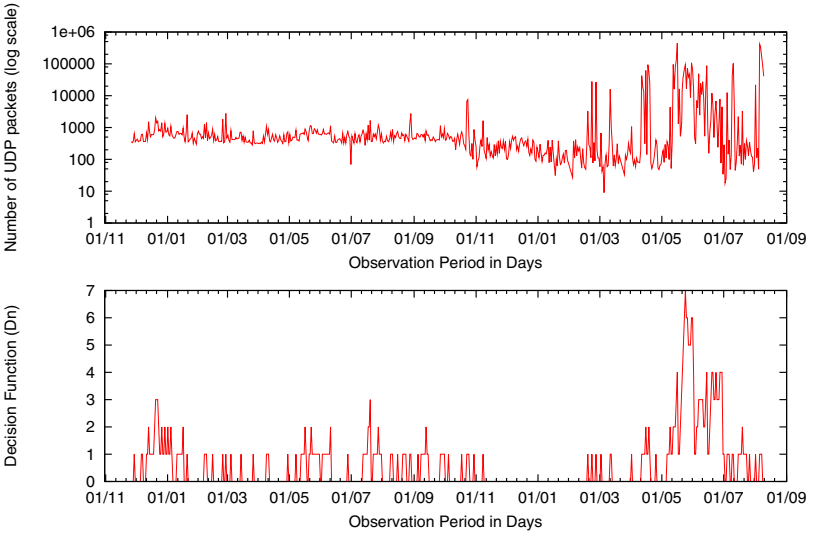


Fig. 3. Change points detected in primary parameter (UDP traffic)

described in Section 4 the change point identified in d^k_{udp} is not significant (less than 30On the other hand the change point identified in $P^{k,r}_{udp}$ is not only significant but also correlated with change in primary parameter. This change point was due to an increase in the UDP traffic on four destination ports 1030-1033 and 1434 collectively receiving more than 50UDP traffic. During this change point no significant activity either by specific source/dest ination address or on specific destination port was observed which was confirmed by the lack of change point in these secondary parameters.

The first nested change point in N^k_{udp} was observed from 12/12/2006 to 5/01/2007. Extracting useful information related to this change point using current correlation algorithm is a difficult task due to the complex correlation of change points observed in secondary parameters. During this period change points in 6 secondary parameters were identified with no change continued for the whole duration. Currently analysis of such complex correlation patterns is not considered and is part of our future work.

The second nested change point was observed from 11/01/2007 to 17/01/2007. During this period nested change was observed on the last day. This change point was correlated with change points in p^k_{udp} and $P^{k,r}_{udp}$. Observe that correlated nested change point was identified in both p^k_{udp} and $P^{k,r}_{udp}$, both observe nested change point on 17/01/2007. The simultaneous occurrence of these nested change points in secondary parameters, p^k_{udp} and $P^{k,r}_{udp}$, validates the nested change point identified in primary parameter N^k_{udp}. This change was due to the increased MS SQL slammer activity on port 1434 which received more than 60The nested change was also due to an increase in the traffic on the same port. Even though the same destination port was targeted in both changes, the simultaneous

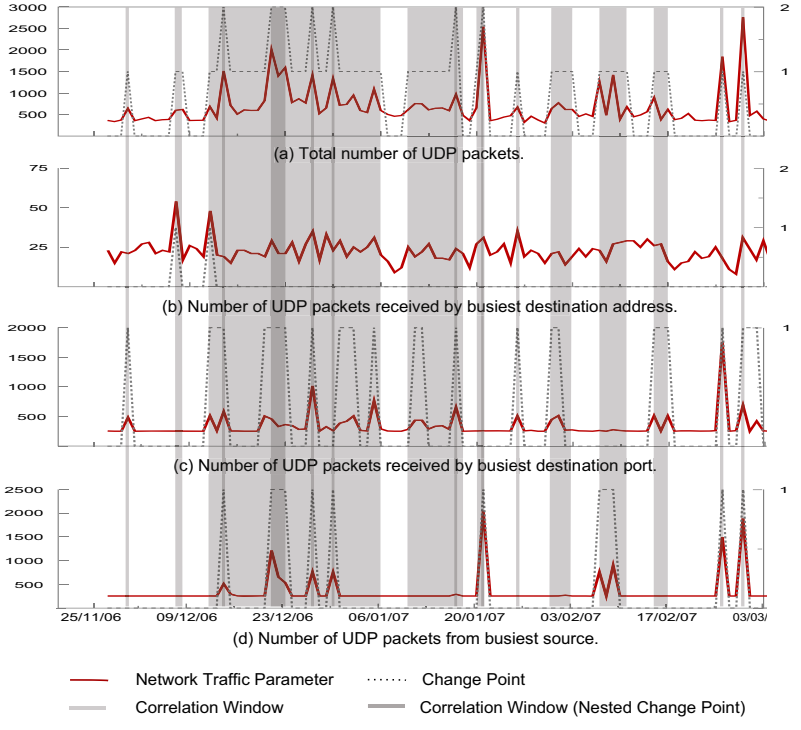


Fig. 4. Change point correlation (UDP traffic)

occurrence of nested changes in primary and secondary parameters alerts to the significant change in UDP traffic on port 1434 during pre-nested and nested changes, traffic on port 1434 is almost doubled during nested change.

An interesting correlation was observed for change point identified in primary parameter on 12/03/2008 which continued for the next day. During this period 8 out of 9 secondary parameters were observed to have correlated change point with primary parameter. In addition nested change point was observed in both $P^{k,w}_{udp}$ and p^{k}_{udp} whereas no nested change point was observed in the primary parameter. This was due to the fact that the change detection algorithm only identifies nested change if there is a significant increase in the parameter value. Although there was significant increase in both $P^{k,w}_{udp}$ and p^{k}_{udp}, an increase in the primary parameter was not observed. In fact the primary parameter was decreased from over 15000 UDP packets to just 1086 packets during that time. In addition the change point in the other 4 secondary parameters was only for one day and was not observed on the second day of the change in primary parameter. During the analysis it was observed that the change on 12/03/2008 was due to the increased activity on a single destination port 13276 on destination address x.x.x.221 which was targeted by the large number of sources. This behavior was not observed on the second day of the change which was confirmed by the lack of change point in S^{k}_{udp}, Y^{k}_{udp}, s^{k}_{udp} and $P^{k,r}_{udp}$.

Although the proposed technique is limited to only simple correlations, our analysis showed some encouraging results. Using the proposed technique we were able to automatically extract and validate more than 60and nested changes, identified in the primary parameter.

7 Conclusion and Future Directions

In this paper, we have proposed a technique for correlating change points among different traffic parameters in order to automatically extract the information related to the anomalous event. The motivation behind our work was to to automate the extraction of information related to change points and to validate the detected change points by correlating change points in primary and different secondary traffic parameters. The applicability and usability of the proposed algorithm is analyzed with the help of real network traces collected from dedicated block of unused IP addresses. It is observed that the proposed technique is not only helpful in automatically extracting information related to detected changes but also can be used to validate the identified changes at the first place.

Even though the proposed technique is limited to only simple correlations, the preliminary results are indeed encouraging. We believe that the algorithm can be improved to detect and validate complex correlations and is part of our future work. In addition we aim to use the proposed technique in automatically generating signatures related to detected change points which can then be used to detect similar behaviors in the future.

References

1. White, G.B., Fisch, E.A., Pooch, U.W.: Computer System and Network Security. CRC Press, Boca Raton (1995)
2. Jiang, X., Xu, D.: Collapsar: a vm-based architecture for network attack detention center. In: Proceedings of the 13th Conference on USENIX Security Symposium, Berkeley, CA, USA, p. 2. USENIX Association (2004)
3. Bailey, M., Cooke, E., Jahanian, F., Nazario, J., Watson, D.: The internet motion sensor: A distributed blackhole monitoring system. In: Proceedings of Network and Distributed System Security Symposium, pp. 167–179 (2005)
4. Harder, U., Johnson, M., Bradley, J.T., Knottenbelt, W.J.: Observing internet worm and virus attacks with a small network telescope. In: Proceedings of the 2nd Workshop on Practical Applications of Stochastic Modelling, June 2005, pp. 113–126 (2005)
5. Moore, D., Shannon, C., Voelker, G.M., Savage, S.: Network telescopes: Technical report. Technical Report tr-2004-04 (July 2004)
6. Vanderavero, N., Brouckaert, X., Bonaventure, O., Charlier, B.L.: The honeytank: a scalable approach to collect malicious Internet traffic. International Journal of Critical Infrastructures 4(1), 185–205 (2008)
7. Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S., Weaver, N.: Inside the slammer worm. IEEE Security & Privacy Magazine 1(4), 33–39 (2003)
8. Rajab, M.A., Monrose, F., Terzis, A.: Worm evolution tracking via timing analysis. In: Proceedings of the ACM workshop on Rapid malcode, pp. 52–59 (2005)

9. Moore, D., Voelker, G.M., Savage, S.: Inferring Internet denial-of-service activity. In: Proceedings of the USENIX Security Symposium, pp. 9–22 (2001)
10. Kumar, A., Paxon, V., Weaver, N.: Exploiting underlying structure for detailed reconstruction of an internet-scale event. In: Proceedings of the ACM Internet Monitoring Conference (2005)
11. Barford, P., Nowak, R., Willett, R., Yegneswaran, V.: Toward a model for source addresses of internet background radiation. In: Proceedings of the Passive and Active Measurement Conference of Stochastic Modelling (March 2006)
12. Fukuda, K., Hirotsu, T., Akashi, O., Sugawara, T.: Correlation among piecewise unwanted traffic time series. In: Proceeding of the IEEE Global Telecommunications Conference, December 2008, pp. 1–5 (2008)
13. Soltani, S., Khayam, S.A., Radha, H.: Detecting malware outbreaks using a statistical model of blackhole traffic. In: Proceeding of the IEEE International Conference on Communications, May 2008, pp. 1593–1597 (2008)
14. Ahmed, E., Clark, A., Mohay, G.: A novel sliding window based change detection algorithm for asymmetric traffic. In: Proceedings of the IFIP International Conference on Network and Parallel Computing, October 2008, pp. 168–175 (2008)
15. Ahmed, E., Clark, A., Mohay, G.: Effective change detection in large repositories of unsolicited traffic. In: Proceedings of the Fourth International Conference on Internet Monitoring and Protection (May 2009)
16. Basseville, M., Nikiforov, I.V.: Detection of abrupt changes: theory and application. Prentice Hall, Englewood Cliffs (1993)
17. Jung, J., Paxson, V., Berger, A.W., Balakrishnan, H.: Fast portscan detection using sequential hypothesis testing. In: Proceedings of IEEE Symposium on Security and Privacy, May 2004, pp. 211–225 (2004)
18. Chan, J., Leckie, C., Peng, T.: Hitlist worm detection using source IP address history. In: Proceedings of Australian Telecommunication Networks and Applications Conference (2006)
19. Bo, C., Fang, B.-X., Yun, X.-C.: A new approach for early detection of internet worms based on connection degree. In: Proceedings of International Conference on Machine Learning and Cybernetics, August 2005, vol. 4, pp. 2424–2430 (2005)
20. Wang, H., Zhang, D., Shin, K.G.: Change-point monitoring for the detection of DoS attacks. IEEE Transactions on Dependable and Secure Computing 1(4), 193–208 (2004)
21. Chen, W., Yeung, D.-Y.: Defending against tcp syn flooding attacks under different types of ip spoofing. In: Proceedings of the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies, April 2006, p. 38 (2006)
22. Tartakovsky, A.G., Rozovskii, B.L., Blazek, R.B., Kim, H.: A novel approach to detection of intrusions in computer networks via adaptive sequential and batch-sequential change-point detection methods. IEEE Transactions on Signal Processing 54(9), 3372–3382 (2006)
23. Siris, V.A., Papagalou, F.: Application of anomaly detection algorithms for detecting SYN flooding attacks. Computer Communications 29(9), 1433–1442 (2006)
24. Thottan, M., Ji, C.: Anomaly detection in IP networks. IEEE Transactions on Signal Processing 51(8), 2191–2204 (2003)
25. Hajji, H.: Statistical analysis of network traffic for adaptive faults detection. IEEE Transactions on Neural Networks 16(5), 1053–1063 (2005)
26. Hajji, H.: Baselining network traffic and online faults detection. In: IEEE International Conference on Communications, vol. 1, pp. 301–308 (2003)

27. Bu, T., Chen, A., Wiel, S.V., Woo, T.: Design and evaluation of a fast and robust worm detection algorithm. In: Proceedings of the 25th IEEE International Conference on Computer Communications, April 2006, pp. 1–12 (2006)
28. Limthong, K., Kensuke, F., Watanapongse, P.: Wavelet-based unwanted traffic time series analysis. In: International Conference on Computer and Electrical Engineering, December 2008, pp. 445–449 (2008)
29. Tartakovsky, A.G., Veeravalli, V.V.: An efficient sequential procedure for detecting changes in multichannel and distributed systems. In: Proceedings of the Fifth International Conference on Information Fusion, vol. 1, pp. 41–48 (2002)
30. Salem, O., Vaton, S., Gravey, A.: A novel approach for anomaly detection over high-speed networks. In: Proceedings of the European Conference on Computer Network Defense. INFO - Dépt. Informatique (Institut TELECOM; TELECOM Bretagne) (October 2007)
31. Idé, T., Inoue, K.: Knowledge discovery from heterogeneous dynamic systems using change-point correlations. In: Proceedings of the SIAM international conference on data mining (2005)
32. Agarwal, M.K., Gupta, M., Mann, V., Sachindran, N., Anerousis, N., Mummert, L.: Problem determination in enterprise middleware systems using change point correlation of time series data. In: Proceedings of the 10th IEEE/IFIP Network Operations and Management Symposium, April 2006, pp. 471–482 (2006)
33. Kaplan, A.Y., Shishkin, S.L.: Application of the change-point analysis to the investigation of the brain electrical activity. In: Brodsky, B.E., Darkhovsky, B.S. (eds.) Nonparametric Statistical Diagnosis: Problems and Methods, pp. 333–388 (2000)

An Improved Attack on TKIP

Finn M. Halvorsen, Olav Haugen, Martin Eian, and Stig F. Mjølsnes

Dep. of Telematics
Norwegian University of Science and Technology
Trondheim, Norway

Abstract. Beck and Tews described the first practical cryptographic attack on IEEE 802.11i TKIP in November 2008, and this paper continues this line of protocol cryptanalysis. We show that their attack on TKIP can be used to create an ARP poisoning attack and a cryptographic DoS attack. Moreover, we are able to decrypt DHCP ACK packets, which are over 12 times longer than the ARP packet used by Beck and Tews. Our method of analysis recovers 596 bytes of keystream that can be used in new attacks on other control protocol messages.

1 Introduction

1.1 Motivation

Today, wireless networks are so widely deployed that they have become almost ubiquitous. The convenience of installing a local area wireless network without having to worry about cables overweigh the fact that wireless networks also are prone to become a security risk if not properly configured. The security mechanism of Wired Equivalent Privacy (WEP) was developed in order to secure wireless networks and provide security equivalent to the one that could be expected from a wired network. The development of wireless security protocols have been a race between the IEEE 802.11i/WiFi Alliance groups developing security standards and hackers developing attacks against those standards. This is clearly illustrated as a timeline in Figure 1.

When WEP failed miserably [4,5,8,7] to deliver the required security, the IEEE 802.11i Temporal Key Integrity Protocol[1] (TKIP) [3] was built around WEP to fix its flaws and provide backward compatibility with older equipment. Much resources and money were invested into upgrading old WEP networks to TKIP.

The TKIP mechanism has been widely deployed and considered to be a secure upgrade of WEP until Beck and Tews, in November 2008, described an attack against TKIP [1]. Even though their attack proved to be limited, the new techniques may open new paths for serious attacks.

The motivation for our research is based on the fact that we believe this is only the beginning in discovering weaknesses in TKIP. Wireless security is an exciting field of study, and we aim to find more weaknesses and new application

[1] The WiFi Alliance refers to TKIP as WPA(1).

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 120–132, 2009.

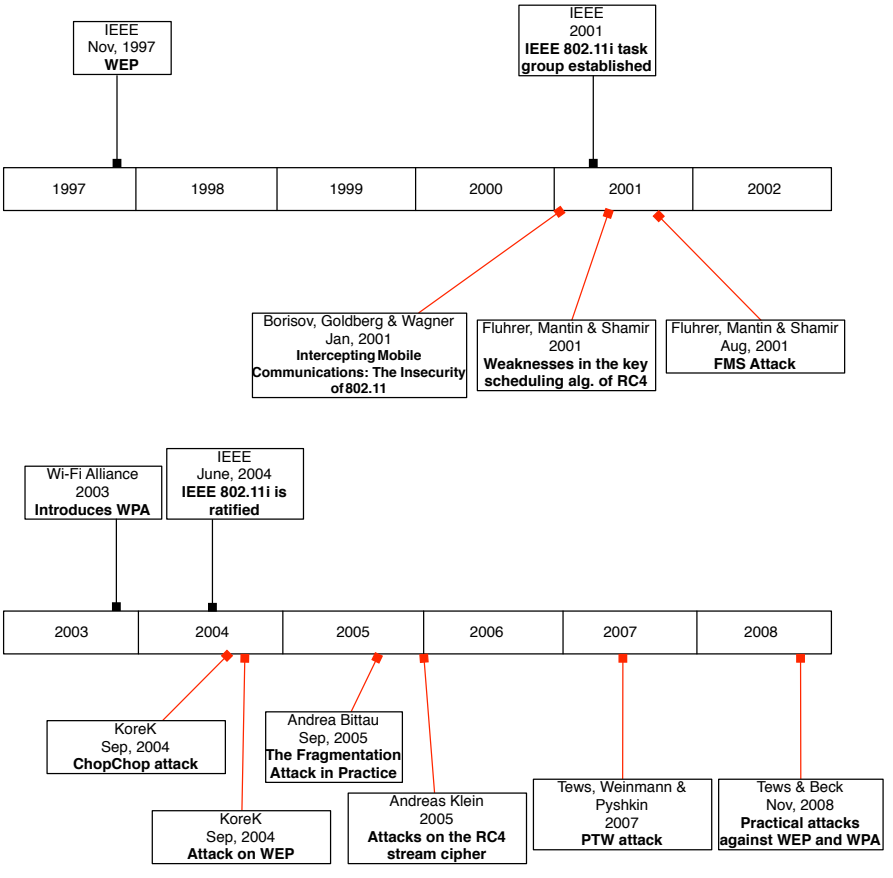


Fig. 1. A timeline of the development of wireless security compared with the development of attacks and discoveries of vulnerabilities

areas of the attacks on TKIP. We hope that our work may contribute to motivate people to migrate their wireless security protocols to the more secure alternative Counter mode with CBC-MAC Protocol (CCMP).

1.2 Problem Description and Results

The goal for our research was to study the attack by Beck and Tews in detail and look for new application areas. Additionally, we aimed to enhance the original attack by Beck and Tews, by looking for other weaknesses in both the TKIP protocol itself, as well as other protocols that are used in wireless networks. As a result, we were able to decrypt a Dynamic Host Configuration Protocol (DHCP) Acknowledgment (ACK) packet, which yields 598 bytes of keystream. This is 12.4 times more than the original attack by Beck and Tews. This is further described in Section 3.

1.3 Outline of the Paper

The structure of this paper is as follows:

- Section 2 gives a short description of Beck and Tews original attack on TKIP.
- Section 3 gives a detailed description of our improved attack on TKIP that is able to decrypt a DHCP ACK packet.
- Section 4 presents the performance of the attack.
- Section 5 suggests further work that should be investigated on this subject.
- Section 6 states the conclusions of the paper.

2 Beck and Tews' Attack on TKIP

In November 2008, Martin Beck and Erik Tews released a paper titled *Practical Attacks Against WEP and WPA* [1], together with *tkiptun-ng*, a new tool in the Aircrack-ng suite[2]. In addition to an enhanced version of the PTW attack on WEP, they released a modified version of the Chopchop attack [2] directed against the TKIP protocol as well. This section will give a short overview of Beck and Tews's attack on TKIP.

2.1 TKIP Overview

TKIP had one important design goal; it should be implementable on old WEP hardware. Because of this limitation the protocol still uses WEP encapsulation, but was designed to provide additional protection against all known attacks on WEP. Hence, the stream cipher RC4 is still used to encrypt the data and the CRC-32 Integrity Check Value (ICV) is still appended to the plaintext before encryption.

The 802.11 2007 standard [3] defines four modifications of WEP that is made by TKIP.

- The use of a new Message Integrity Check (MIC), which is generated by the keyed cryptographic algorithm Michael.
- The MIC is, because of the design constraints, not very secure. Therefore TKIP implements additional countermeasures to handle this.
- Replay protection, with the use of a per-MPDU TKIP sequence counter (TSC).
- TKIP uses a cryptographic per-packet key mixing function to defeat weak-key attacks against the WEP key.

2.2 TKIP Countermeasures

The designers of TKIP realized that Michael was not sufficiently secure. As a consequence, they implemented some countermeasures. The countermeasures are

[2] Tkiptun-ng tool: `http://www.aircrack-ng.org/doku.php?id=tkiptun-ng`

designed to prevent an attacker from trying to crack the MIC by using brute force. This feature of TKIP is explained in detail because it is an essential part of Beck and Tews' attack on TKIP [1]. The IEEE 802.11 2007 standard specifies [3] how STAs and APs shall react on MIC failures, and suggests that such events *should* be logged and *must* be kept below two per minute. The last restriction implies that if two MIC failures are detected within one minute, a STA or AP must activate the TKIP countermeasures. A MIC failure occurs when a received packet has a valid ICV but an invalid MIC. When the countermeasures are activated, the AP will delete all temporal keys and shut down all TKIP traffic for one minute. After this minute has passed, all STAs will have to re-authenticate and create new temporal keys. This will give the attacker one try per minute on guessing the right MIC, making it infeasible for the attacker to guess the correct value. In this way the WEP ICV helps to prevent false detection of MIC failures, and prevents the use of countermeasures when no attack is taking place [3].

2.3 Attack Requirements

A set of conditions must be met in order to mount the attack on TKIP. First of all, the attack relies on IEEE 802.11e Quality of Service (QoS) [3], sometimes referred to as WiFi Multimedia (WMM), to be enabled. Each QoS channel maintains its own TSC. The TSC is only incremented if a valid packet is received, and a packet is accepted only if the TSC value is higher than the last received packet's TSC value. This makes it possible to capture a packet on one QoS channel and inject it into another QoS channel with a lower TSC value. In most QoS enabled networks, regular data is sent on channel 0. This means that all other channels most likely have a lower TSC value, and as a result, can be vulnerable to a Chopchop like attack. The number of QoS channels available varies from four to 16 among different implementations. This number limits the number of packets (3-15) the attacker is able to inject into the network.

Another requirement for the attack to succeed is the length of the key renewal interval in TKIP. The attack procedure must wait one minute between each byte chopped in order to avoid triggering the MIC countermeasures [3]. If activated, the AP will shut down all TKIP traffic for one minute followed by a key renewal. Therefore the attack will be bound to use more time than the original Chopchop attack on WEP. If a key renewal occurs while the attack is being executed, the attack will fail and it must start over again. Hence, the key renewal interval needs to be longer than the attack time. It is common practice by access point manufactures to set this interval to 3600 seconds, which is approximately four times longer than the attack needs to complete.

2.4 How the Attack Works

The attack on TKIP enables the attacker to decrypt an access point-to-station (AP-to-STA) Address Resolution Protocol (ARP) request. By doing this, the attacker will obtain the keystream for that packet as well as the MIC key for

AP-to-STA communication, which can be used to create and inject custom AP-to-STA packets into the network.

The attack can be summarized into four different stages:

1. Client de-authentication
2. Modified Chopchop attack
3. Guessing and validating the remains of the packet
4. Reversing the MICHAEL algorithm.

The novelty of this attack is the modified Chopchop attack. The modified Chopchop attack works by "chopping" off the last byte of the packet, the same way that the conventional Chopchop attack [2] works. In contrast to the regular Chopchop attack where traffic is directed to the AP, the modified Chopchop attack acts as an AP sending data to a STA on a QoS channel with a lower TSC than channel 0. The reason for why the attack can only be executed towards a STA is because the STAs are the only entities that send MIC failure report frames. The receiver will silently discard any packet that has an incorrect ICV and incorrect MIC. However, when the correct byte is guessed, the ICV of that packet will be correct, while the MIC will be incorrect. This will cause the STA to send a MIC failure report, indicating that the last guess was correct. As two MIC failure reports within a minute will trigger the MIC countermeasures in the AP, the attacker would need to wait for 60 seconds before chopping the next byte.

Upon decrypting the ICV and MIC value of an ARP packet, the remaining part of the ARP packet can be guessed. This is due to the fact that the MAC addresses of the ARP packets are sent in the clear as a part of the 802.11 headers, while the remaining unknown IP addresses of the ARP packet can be guessed to IP addresses of a local area network. Each guess can be checked by generating the ICV checksum and compare to the decrypted ICV.

The last step of the attack is to reverse the MICHAEL algorithm to obtain the MIC key. The MICHAEL algorithm was never designed to be a one-way function with the same strength as a cryptographic hash function. In fact, it turns out that it is possible, given the MIC and plaintext, to reverse the algorithm as fast as one can do a forward calculation. Thus, by reversing the MIC algorithm, the MIC key can easily be retrieved.

2.5 Application Areas

Beck and Tews suggested several application areas of the attack in their paper, but none of these were implemented in their program. Attacks suggested were based on ARP poisoning, i.e., injection of fake ARP packets into the network. We have developed[3] two attack modes based on Beck and Tews' implementation: an ARP poisoning attack and a cryptographic denial-of-service (DoS) attack.

[3] Source code and explanation can be found at
http://forum.aircrack-ng.org/index.php?topic=5876.0

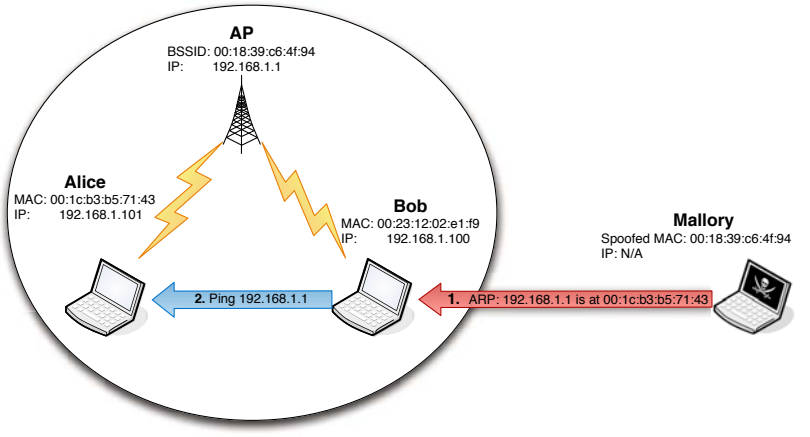


Fig. 2. ARP poisoning attack - The attacker injects a fake ARP reply to corrupt the STA's ARP cache

The ARP poisoning attack works by creating a fake ARP packet and using the obtained keystream from the attack on TKIP to inject it into the wireless network on one of the unused QoS channels. As illustrated in Figure 2, this will cause the ARP cache of a computer to be corrupted, which will disrupt the routing on the network.

Additionally, we created a cryptographic DoS attack by modifying the Beck and Tews attack. This obvious attack was not described in Beck and Tews' original paper. In the original code, the attacker would wait for 60 seconds upon receiving a MIC failure report to avoid triggering the MIC countermeasures. However, the goal of a DoS attack would be to do the exactly opposite, namely to trigger the MIC countermeasures. Hence, upon receiving a MIC failure report, the attacker will just inject the same packet that triggered the MIC failure report once more. This will cause the AP to shut down and re-key. It requires only 128 packets on average to trigger the first MIC failure report, and one more packet to trigger the MIC countermeasures, in total 129 packets on average. The attacker must wait one minute for the AP to restart before re-initiating the attack.

3 The Improved Attack on TKIP

3.1 DHCP ACK Packet

The Beck and Tews attack is limited to decrypting AP-to-STA ARP packets. Since ARP packets are short, the obtained key stream is short and thus limited to injection of ARP packets only. This section will present a way of decrypting larger DHCP ACK packets, with typically size in the range of 330 to 584 bytes. This will enable attacks using injection of longer packets. This improved attack is not a re-implementation, but rather an extension of the code of the *tkiptun-ng* tool. We must emphasize that it is intended as a proof-of-concept attack and is

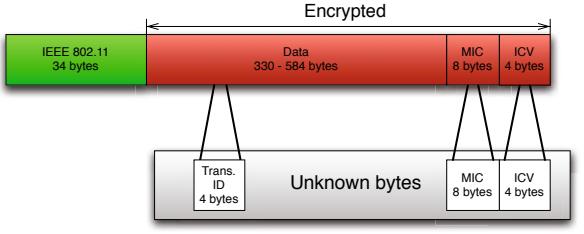


Fig. 3. An encrypted DHCP ACK packet with 16 unknown bytes

not designed to work with a generic set of equipment. The attack is still only limited to injection of AP-to-STA packets, the enhancement being the injection of a wider variety of much longer packets. The same requirements as described in Section 2.3 apply for this attack.

Typically, a client will send DHCP and ARP requests when configuring for a new network. DHCP ACK messages are sent as confirmations to a DHCP request. When a STA has been disconnected from the network (i.e. de-authenticated by the attacker) and tries to reconnect to the network, it will typically send a DHCP request for the same IP address it was previously assigned. In most cases, the AP will then respond with a DHCP ACK to acknowledge the request. By inspection, we discovered that a DHCP ACK message mostly contains known plaintext, even though it can be up to 584 bytes in size. The reason for this is the extensive use of 0-padding, which in many cases make up most of the data in the packet.

Further investigation of the DHCP ACK determined that the format of these messages to be manufacturer specific. This means that e.g., a Linksys router will respond with the same format of this message, while another manufacturer may respond differently. It is, however, possible to overcome this problem by looking at the Basic Service Set Identifier (BSSID) of the AP. The BSSID yields information about the manufacturer, which in combination with a database on how different router manufacturers format their DHCP ACKs could give a good indication of the format, length, options and IP ranges of packets coming from a specific AP/router.

Given that we know the manufacturer specific format for the DHCP ACK, the only bytes that cannot be determined are the IP addresses, the Transaction ID, the MIC and the ICV. By running Beck and Tews' attack [1] on an ARP packet first, we would obtain the IP addresses of the STA and AP, which could be further used as known plaintext in the DHCP ACK packet. Now, we have a remaining 16 unknown bytes, as illustrated in Figure 3.

3.2 How the Attack Works

The major difference between the original attack and our extension is that while the entire data part of an ARP packet can be guessed, only parts of a DHCP ACK packet can be guessed. The DHCP ACK packet contains an unknown Transaction ID field in the middle of the packet. As this field is 32 bits in length, guessing

this field is infeasible. This means that rather than performing a modified Chopchop attack followed by guessing the remains of the packet, we must simulate a modified Chopchop attack in order to keep the state of the program up-to-date after inserting known bytes. We refer to this function as the `simulate_chopchop` function. This function merely assumes that a correct guess has been made, and updates the state of the program the same way as when a regular correct guess has been made. At this point, since we know the bytes, we can skip the communication with the STA and the 60-second waiting delay between each packet.

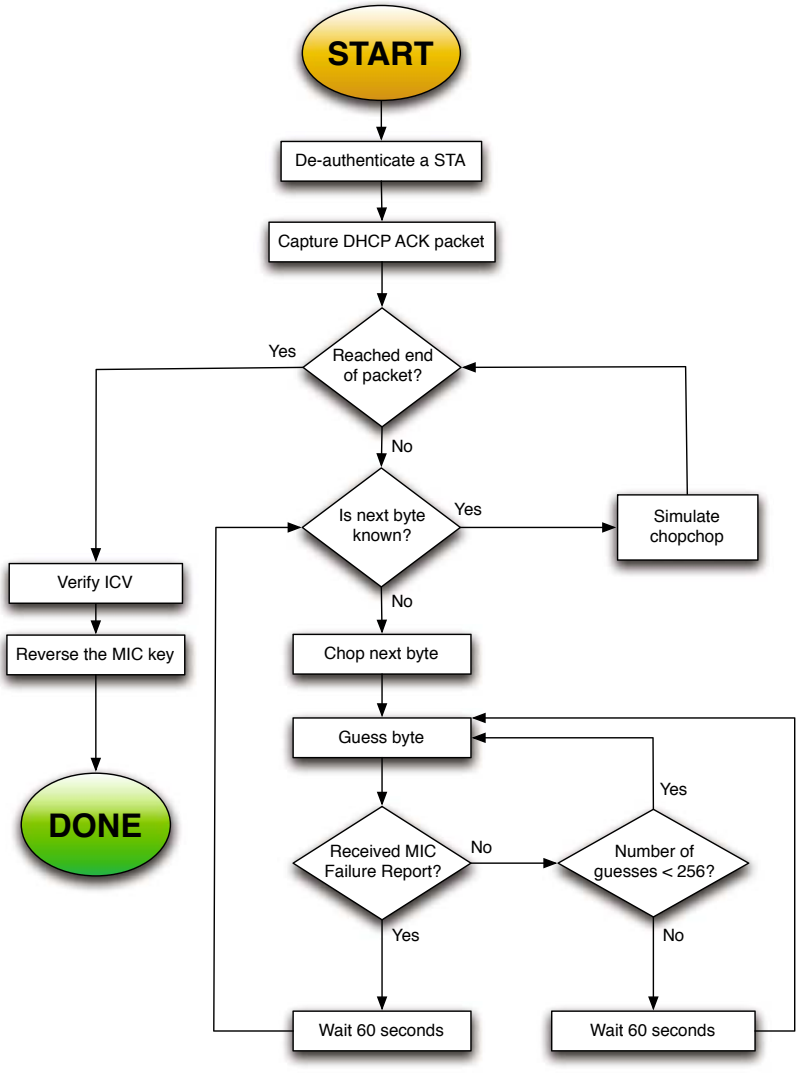


Fig. 4. A flowchart of our improved attack on TKIP

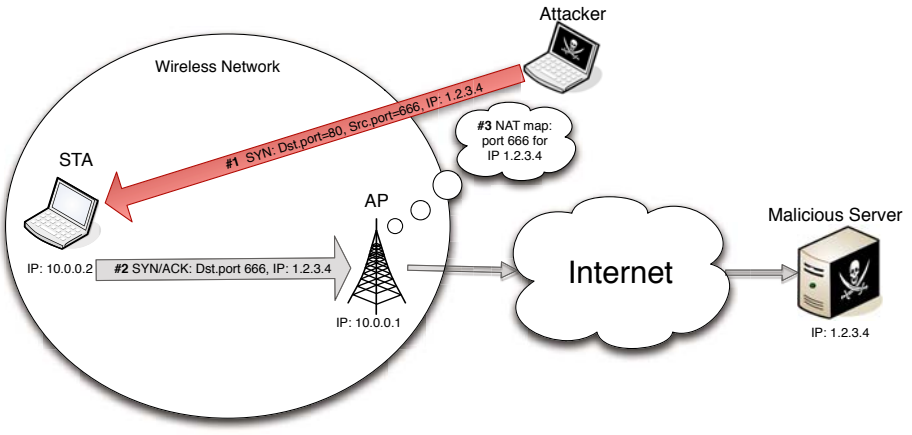


Fig. 5. NAT traversal attack using TCP SYN packets to open a port in the firewall of the router, allowing external machines to communicate with a machine on the internal network

This `simulate_chopchop` function demands very little processing power, and completes in a negligible amount of time. Additionally, the IP and UDP header checksums must be calculated and inserted at the appropriate positions. Figure 4 shows a flowchart, explaining the operation of the program.

3.3 Consequences

In our experimentation, we were able to obtain 596 bytes of keystream from a Linksys WRT54GL Wireless router. 596 bytes of keystream are significantly more (12.4x more) than the 48 bytes of keystream recovered from the original attack by Beck and Tews [1]. While their attack was limited to inject ARP packets only, with 596 bytes of keystream the possibilities become overwhelming. Now, it is possible to inject almost all kinds of traffic concerning control information such as TCP SYN/ACK, DNS, DHCP, ICMP, ARP and more.

A possible attack could be to inject fake DHCP ACK packets containing a fake Domain Name System (DNS) Server Address. This will cause the receiving client to perform DNS queries to a DNS Server of the attacker's choice. First, a client must have sent a DHCP request, otherwise the client will reject incoming DHCP ACKs. We discovered that by causing an IP conflict on the local area network, some clients[4] would initiate a DHCP renewal by sending a DHCP request. If the DHCP transaction ID of this packet can be determined, a fake DHCP ACK response with the same DHCP transaction ID could be injected before the DHCP server has time to respond. Our experimentation[4] showed that the attacker would need to inject four packets in order to create an IP conflict at a STA, by sending fake gratuitous ARP requests to the STA.

[4] We observed this behavior on Mac OS X 10.5.6.

Another attack that seems possible when limited to injection of AP-to-STA packets only, is a Network Address Translation (NAT) Traversal attack, as illustrated in Figure 5. The idea behind this attack is to inject a fake TCP SYN packet that appears to originate from an external IP address at a specific TCP port. The machine on the internal network will then respond with a TCP SYN/ACK packet, which in turn will force the router to establish a NAT mapping between the internal and external ports and IP addresses. The external machine will then receive the TCP SYN/ACK and can act accordingly. The attacker will now be able to send traffic directly to the internal client on the open port in the firewall. This could for instance be used to exploit some un-patched vulnerability at the client. Additionally, this attack will reveal the Internet IP address of the network, which could be useful in other scenarios as well.

4 Attack Performance

The duration of the improved attack is somewhat longer than the original *tkiptun-ng*, but the resulting keystream obtained is more than tenfold that of the original attack. The proof-of-concept implementation needs to chop a total of 16 bytes. This includes the 12 bytes of the original attack, MIC and ICV. In addition to this the attacker needs to chop the four-byte DHCP Transaction ID. These four bytes add approximately 5 minutes to the attack time. The average time to complete this attack without missed MIC failure reports and initialization will then be:

$$\frac{16 \times 128}{10} + 15 \times 60 \approx 18 \; \textit{minutes and} \; 25 \; \textit{seconds}. \tag{1}$$

Where 16 is the number of bytes to chop, 128 is the average number of guesses per byte, 10 represents ten guesses per second, 15 is the number of times to wait between bytes and 60 is the MIC failure interval in seconds.

The simulated chopchop, introduced in our improved attack, takes negligible time to compute. Thus, the real-world time for the improved attack is in the range of 20 to 25 minutes, depending mainly on the number of MIC failure reports missed. Even if this improved attack needs more time to complete, it will recover 596 bytes of keystream compared to 48 bytes in the original attack. This is an increase of obtained keystream by 12.42 times, in only approximately 20% longer time. Although the previous statement is true, the IP addresses of the DHCP server, in most cases the AP, and STA needs to be known in advance of the attack. If this information cannot be determined otherwise then the original tkiptun-ng needs to be used. Hence, the total time of the attack will be closer to 40 minutes.

It is also possible to mount the attack if additional bytes are unknown, for instance the DNS server IP address. This would add about 70 seconds to the total attack time for each unknown byte.

5 Further Work

5.1 DHCP ACK Formats

Focusing on proof-of-concept, we implemented an improved attack that has been tested only with our specific equipment. A more generic attack will require a database of DHCP ACK formats of the manufacturers. It is a simple task to detect the manufacturer based on the first bytes of the MAC address of the AP. It could also be possible to detect the model if a certain model is limited to a certain MAC address range.

5.2 Obtaining Two-Way Keystream

Obtaining keystream for both directions still remains a challenge. This is one of the limitations to both our improved and the original attack by Beck and Tews. The reason why the attack is limited to AP-to-STA keystream is that only STAs send MIC Failure report frames. This means that if a chopchop attack were mounted against the STA-to-AP keystream, the attacker would have no way of knowing when a correct guess was made. Beck and Tews suggest that it would be possible to mount such an attack if the EAPOL handshake used the same random nonces for every re-keying [1]. This implies that both the Supplicant (STA) and Authenticator (AP) have a flawed implementation of TKIP, and for this reason a very unrealistic scenario.

Given that the MIC key was identical for both directions, or that the attacker somehow was in possession of both keys, the keystream for both directions could easily be obtained. This could be achieved by sending a packet with a known reply to the STA, for instance an ICMP Ping. The answer could then be XORed with the known reply, including the calculated MIC and ICV, resulting in keystream for STA-to-AP communication. This type of known reply attack could then be repeated indefinitely, giving the attacker an unlimited number of useable keystreams.

5.3 Fragmentation Attack

The fragmentation attack [6] is a powerful attack against WEP. A similar approach could work against TKIP as well. If an attacker gets hold of two keystreams for different TSCs, by performing two consecutive *tkiptun-ng* attacks, it should be possible for the attacker to send a packet in two fragments, one with each keystream. These fragments should be sent on the same QoS channel. Thus, the attacker would be able to send a packet twice the size of one keystream.

It might also be possible to fragment across QoS channels, although this is less likely to work. Then an attacker would only need to obtain one keystream, and send the fragments on different QoS channels. If the attacker could get hold of keystream and MIC key for both directions, he would be able to mount an attack similar to the fragmentation attack on WEP. The attacker could send fragmented packets with known replies, and thus obtain longer keystream for each consecutive injection. The attacker would then be able to acquire 1500 bytes (maximum transmission unit) of keystream for both directions.

5.4 Key Recovery Attack

The ultimate attack on wireless networks is a key recovery attack. Such an attack will reveal the pre-shared key, and allow the attacker to communicate as a legitimate STA on the network. The attacker will also be able to decrypt the traffic of all other STAs on the network.

Currently, the only way to obtain the pre-shared key on a TKIP or CCMP secured wireless network, is to mount a brute force or dictionary attack on the EAPOL handshake. Such attacks are successful against weak passwords by using a dictionary attack. Recent advances in Graphics Processing Unit (GPU) technology have also made it possible to utilize the immense computational power of such chips. This has made it possible to successfully attack even relatively strong keys in viable time.

The original *tkiptun-ng* attack and our improved attack do not target the pre-shared key (PSK). However, the attacks do obtain the MIC key for AP-to-STA communication. This key is a part of the Pairwise Transient Key (PTK) derived from the pre-shared key in the EAPOL handshake. Thus, the attacker is in possession of 64 bits of the 512-bit PTK. The attacker is now in possession of both some of the input and parts the output of the transient key computation algorithm (PRF-512). It might be possible to reduce the time needed for an attack on the PMK if this known MIC key could in some way be included in the attack algorithm. This would require cryptanalysis of the underlying PRF-512, which utilizes the hash function SHA-1. The possibilities for this should definitely be investigated further. By repeating the attack, the attacker would be in possession of additional MIC keys for a given handshake. The statistical relationship between these should also be explored.

6 Conclusion

Until recently, WEP was the only wireless protocol with exploitable weaknesses. TKIP and CCMP were considered cryptographically strong, i.e. the only known weakness would be the use of weak or easily guessed passwords. In November 2008, Beck and Tews presented a breach in the TKIP's security, which allowed an attacker to decrypt small ARP packets. Although the exploitable effect of their analysis was limited, nevertheless, it was the first practical attack on TKIP.

The Beck and Tews attack is still, as of May 2009, very experimental, and we have experienced changing success rate with different system configurations. We were able to extend the *tkiptun-ng* to function as an ARP poisoning attack and a cryptographic DoS attack. Additionally, we have created a TKIP attack that can decrypt DHCP ACK packets. This is more than 12 times the size of an ARP packet, which is the target of the Beck and Tews attack. This opens up for new application areas for the attack. Though not yet implemented, we have described a theoretically possible attack that will spoof the DNS server IP address of a client. An attacker could make a client communicate with a malicious DNS server rather than the desired one, and thus easily perform a *phishing* attack.

There are some requirements for the attack to succeed. QoS/WMM must be enabled at the AP, and the key renewal interval must be longer than the attack time. These conditions are frequently met in a real network, and as a consequence, the attack should be considered a real threat.

TKIP was developed to fix the insecurity of WEP, but now it is TKIP that needs to be fixed. We begin to see driver updates fixing this issue, like for instance OpenBSD, which has patched its client stack to counter this attack. Nevertheless, the TKIP design was constrained by WEP, and not surprisingly it inherits some of its weaknesses. Instead of fixing TKIP, we believe it is time to move on and start migrating to CCMP.

The authors do acknowledge that the initial intended five-years life span of TKIP, now - seven years into existence - has expired. Even so, due to the widespread use of the protocol, attacks on TKIP should not be ignored.

References

1. Beck, M., Tews, E.: Practical attacks against WEP and WPA. Cryptology ePrint Archive Report 472, 79–86 (2008)
2. KoreK: chopchop (Experimental WEP attacks) (2004), http://www.netstumbler.org/f50/chopchop-experimental-wep-attacks-12489/
3. IEEE Standard for Information technology-Telecommunications and information exchange between systems-Local and metropolitan area networks-Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999) (2007)
4. Fluhrer, S., Mantin, I., Shamir, A.: Weaknesses in the key scheduling algorithm. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 1–24. Springer, Heidelberg (2001)
5. Stubblefield, A., Ioannidis, J., Rubin, A.D.: A key recovery attack on the 802.11b wired equivalent privacy protocol (WEP). ACM Trans. Inf. Syst. Secur. 7, 319–332 (2004)
6. Bittau, A., Handley, M., Lackey, J.: The Final Nail in WEP's Coffin Security and Privacy. In: IEEE Symposium on, pp. 386–400 (2006)
7. Tews, E., Weinmann, R.-P., Pyshkin, A.: Breaking 104 bit WEP in less than 60 seconds. Cryptology ePrint Archive 120 (2007)
8. KoreK.: Next generation of WEP attacks (2004), http://www.netstumbler.org/showpost.php?p=93942&postcount=35

ContikiSec: A Secure Network Layer for Wireless Sensor Networks under the Contiki Operating System

Lander Casado and Philippas Tsigas

Department of Computer Science and Engineering, Chalmers University of Technology,
SE-412 96 Göteborg, Sweden
landerc@student.chalmers.se, tsigas@chalmers.se

Abstract. In this paper we introduce ContikiSec, a secure network layer for wireless sensor networks, designed for the Contiki Operating System. ContikiSec has a configurable design, providing three security modes starting from confidentiality and integrity, and expanding to confidentiality, authentication, and integrity. ContikiSec has been designed to balance low energy consumption and security while conforming to a small memory footprint. Our design was based on performance evaluation of existing security primitives and is part of the contribution of this paper. Our evaluation was performed in the Modular Sensor Board hardware platform for wireless sensor networks, running Contiki. Contiki is an open source, highly portable operating system for wireless sensor networks (WSN) that is widely used in WSNs.

Keywords: Wireless Sensor Networks Security, Link Layer Security.

1 Introduction

Wireless sensor networks (WSNs) are unique networked systems, composed of wireless sensor nodes deployed en masse. WSNs distinguish themselves from other traditional wireless networks by relying on extremely constrained resources such as energy, bandwidth, and the capability to process and store data [1]. The number of applications that make use of WSNs is constantly increasing. The range of these applications goes from military and societal security to habitat and environment monitoring.

For many applications it is essential to provide secure communications. In general, WSNs face the same security risks as conventional wired or wireless networks; eavesdropping, packet injection, replay and denial of service attacks are some of the common attacks in WSNs. Due to the inherent properties of sensor nodes, traditional security protocols are not suitable for WSNs. A set of different attempts to implement secure communication specifically for WSNs appeared recently in the literature, such as TinySec [2], SenSec [1], MiniSec [3], and TinyECC [4]. All of these are designed to run under TinyOS [5], a widely used operating system for sensor nodes.

In 2004 Contiki was presented. Contiki is an open source, highly portable, multitasking operating system for memory-efficient networked embedded systems and wireless sensor networks [6]. Contiki has become quite popular and is attaining a good position in the WSNs community. Contiki does not offer any confidentiality or

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 133–147, 2009.

authenticity services for the communication between nodes. The current version provides only cyclic redundancy checks (CRC-16) to achieve integrity.

In this paper we present ContikiSec, which is the first attempt, to the best of our knowledge, to implement a secure network layer for wireless sensor network architecture for Contiki. ContikiSec has a configurable design, providing three security modes. ContikiSec offers confidentiality, authentication, and integrity in communications under the Contiki operating system. We base our design on existing security primitives that have been proven to provide security properties. We designed a complete solution after careful performance characterization and analysis of the existing security primitives that we selected. Our design tries to balance low energy consumption and security for the Contiki OS. Our design was guided through a set of performance evaluations in the MSB-430 platform, a Modular Sensor Board hardware platform created by ScatterWeb. During this evaluation, which is part of the contribution of the paper and is introduced in Section 4, we present the performance and resource consumption of six different block ciphers in our network. The six block ciphers that we considered are: AES, RC5, Skipjack, Triple-DES, Twofish, and XTEA (please refer to Section 4 for the detailed definitions of these block ciphers). The parameters that we focused on were the encryption speed, memory usage, and energy consumption. Our results show that AES is the most suitable block cipher for the WSNs under consideration. Consequently, we evaluate the performance of the CBC–CS, CMAC, and OCB modes of operation (please refer to Section 4 for the detailed definitions of these modes of operation). Based on the findings of our evaluation results we then present and evaluate the design of ContikiSec.

The rest of the paper is organized as follows. In the next section, we describe the security properties of a secure network layer for wireless sensor networks and we review the related work in WSNs. In the same section, we briefly describe the characteristics of the Modular Sensor Board hardware platform and the Contiki OS. In Section 3 we describe the performance and measurement methodology that we used to evaluate and analyze security primitives for our networks. The evaluation of the selected security primitives (block ciphers and modes of operation) are described in Section 4. In Section 5 we present the ContikiSec design, guided by the evaluation presented in Section 4. Finally, the paper is concluded in Section 6.

2 Background

2.1 Security Properties

The security properties that should be provided by a secure network layer for wireless sensor networks are described below.

Confidentiality. Confidentiality is a basic property of any secure communication system. Confidentiality guarantees that information is kept secret from unauthorized parties. The typical way to achieve confidentiality is by using symmetric key cryptography for encrypting the information with a shared secret key. Symmetric key algorithms could be divided into stream ciphers and block ciphers. In the case of block ciphers, a mode of operation is needed to achieve semantic security.

Semantic Security. Semantic security guarantees that a passive adversary could not extract partial information about the plaintext by observing the ciphertext [3]. Block ciphers do not hide data patterns since identical plaintext blocks are encrypted into identical ciphertext blocks. Thus, a special mode of operation and an initialization vector (IV) are often used and are needed to provide some randomization. IVs have the same length as the block and are typically added in clear to the ciphertext.

Integrity. Integrity guarantees that the packet has not been modified during the transmission. It is typically achieved by including a message integrity code (MIC) or a checksum in each packet. The MIC is computed by calling a cryptographic hash function that detects malicious altering or accidental transmission errors. Checksums are designed to detect only accidental transmission errors.

Authenticity. Data authenticity guarantees that legitimate parties should be able to detect messages from unauthorized parties and reject them. The common way to achieve authenticity is by including a message authentication code (MAC) in each packet. The MAC of a packet is computed using a shared secret key, which could be the same key used to encrypt the plaintext. In addition to authenticity, MACs also provide integrity.

2.2 Existing Security Architectures for WSNs

In recent years, the increased need of security in WSNs has prompted research efforts to develop and provide security modules for these platforms. These efforts go from simple stream ciphers to public key cryptography architectures.

SPINS [7] is the first security architecture designed for WSNs. It was optimized for resource-constrained environments and it is composed of two secure building blocks: SNEP and μTesla. SPINS offers data confidentiality, two-party data authentication, and data freshness. However, SNEP was unfortunately neither fully specified nor fully implemented [2].

In 2004, TinySec [2] was presented as the first fully implemented link layer security suite for WSNs. It is written in the nesC language and is incorporated in the official TinyOS release. TinySec provides confidentiality, message authentication, integrity, and semantic security. The default block cipher in TinySec is Skipjack, and the selected mode of operation is CBC–CS. Skipjack has an 80-bit key length, which is expected to make the cipher unsecure in the near future [8]. In order to generate a MAC, it uses Cipher Block Chaining Message Authentication Code (CBC–MAC), which has security deficiencies [9]. It provides semantic security with an 8-byte initialization vector, but adds only a 2-byte counter overhead per packet. TinySec adds less than 10% energy, latency, and bandwidth overhead.

SenSec [1] is another cryptographic layer for WSNs, presented in 2005. It is inspired by TinySec, and also provides confidentiality, access control, integrity, and semantic security. It uses a variant of Skipjack as the block cipher, called Skipjack-X. It has been conjectured that Skipjack-X is more secure than Skipjack, because it is not vulnerable to brute force attacks. In addition, SenSec provides a resilient keying mechanism.

MiniSec [3] is a secure sensor network communication architecture designed to run under TinyOS. It offers confidentiality, authentication, and replay protection. MiniSec has two operating modes, one tailored for single-source communications, and the other tailored for multi-source broadcast communication. The authors of MiniSec chose Skipjack as the block cipher, but they do not evaluate other block ciphers as part of their design. The mode of operation selected in MiniSec is the OCB shared key encryption mechanism, which simultaneously provides authenticity and confidentiality.

TinyECC [4] is a configurable library for elliptical curve cryptography operations in WSNs. It was released in 2008 and targets TinyOS. Compared with the other attempts to implement public key cryptography in WSNs, TinyECC provides a set of optimization switches that allow it to be configured with different resource consumption levels. In TinyECC, the energy consumption of the cryptographic operations is on the order of millijoules, whereas using symmetric key cryptography is on the order of microjoules [10].

2.3 Sensor Network Platform

Our selected platform is composed of MSB-430 sensor nodes that run the Contiki operating system. The main features of the hardware and the operating system are described below.

Modular Sensor Board (MSB-430). Modular Sensor Board (MSB-430) is a hardware platform for wireless sensor networks created by ScatterWeb [11]. The MSB-430 contains the MSP430F1612 [12] microcontroller, and offers 60 KB of memory divided into 5 KB RAM and 55 KB Flash-ROM. The MSP430F1612 achieves ultra-low power consumption with its five software selectable low power modes of operation.

The MSB-430 is equipped with the Chipcon CC1020 [13] transceiver and a low-noise amplifier. The radio frequency can be selected separately for receiving and transmitting by software. This is an important feature for advanced routing schemes, where multiple radio channels are used. The maximum transmission power is 8.6 dBm, and can be adjusted to reduce power consumption. While the maximum transmission rate is 153.6 kbps, the board is optimized for channel conformity, which allows a typical data rate of 19.2 kbps when using Manchester encoding.

The MSB-430 is supported by the open-source ScatterWeb operating system [11]. In this project the selected operating system is Contiki, which has a port to the MSB-430 platform.

Contiki. Contiki is an open source, highly portable, multi-tasking operating system for memory-efficient networked-embedded systems and wireless sensor networks [6]. Contiki is written in the C language and is designed for microcontrollers with a small amount of memory. A common Contiki configuration uses 2 kilobytes of RAM and 40 kilobytes of ROM. Contiki has been ported to different hardware platforms, such as MSP430, AVR, HC 12, and Z80.

Compared with other operating systems developed for resource-constrained platforms, one of the most interesting features of Contiki is dynamic loading, the ability to load and replace individual applications or services at run-time. Moreover, Contiki

offers a hybrid model with an event-driven kernel where preemptive multi-threading is implemented as an application library [6]. Thus, multi-threading is only used when the application needs it.

3 Performance and Measurement Methodology

Due to the extreme resource limitations of WSNs, it is essential to measure the memory use, encryption speed, and energy consumption of the implemented security primitives. In this section we describe accurate software-based methods to measure parameters such as throughput, ROM, RAM, and energy consumption of WSN applications. These methods are used in our evaluation later on.

3.1 Software Encryption Speed

In order to measure the encryption speed, one way is to use a high precision oscilloscope to check the digital output pin. Another method is to use the Contiki real-time timers. Measuring parameters externally is more complex and often less precise, because the margin of error of the measuring instrument must be taken into account. Therefore, we decided to use the Contiki real-time timers. Certainly, we achieved a very high accuracy using this method.

The auxiliary clock (ACLK) is sourced from a 32768 Hz crystal oscillator. By selecting this clock, we can choose different divisors, such as 1, 2, 4, or 8. Using 1 as a divisor, we obtained the maximum resolution of 30.5176 microseconds. After some experiments, we realized that we can achieve greater resolution by measuring the time that is needed to complete the assignment statement. Using a simple loop, we observed that, on average, a variable could store the same tick value 5.75 times. Based on this observation, we defined the concept of $\mu tick$, which is the time needed to do the assignment statement. Please note that the assignment is always an assignment of a 2-byte value. By using this observation we obtain a resolution of 5.3074 microseconds, an acceptable accuracy compared with related work [14], which has a resolution of 5 microseconds.

3.2 ROM

Due to the limited memory resources of sensor nodes, the ROM-Flash used is an important parameter that has to be considered in the design of a software security layer in WSNs. Conventional cryptographic libraries are too big for embedded devices, the code size of the selected cryptographic suite has to be on the order of a few kilobytes. We are interested in measuring the memory used only by the cryptographic layer and not by the whole program that includes the Contiki operating system. To achieve this, we first compile the code with all the cryptographic functions and measure the size using the *msp430-size* utility [15]. By repeating the same process but without the cryptographic suite we can then calculate the memory required by the cryptographic module by a simple subtraction. This value is the best estimate of the ROM requirement of the security module.

3.3 RAM

RAM memory is even more limited than ROM-Flash. In the MSB-430 nodes, the volatile memory is only 5 KB. This parameter is hard to measure because of the variable size of the stack. The typical way to measure RAM memory is using a real-time debugger and setting breakpoints in the code. Unfortunately, there is no debugger available for the MSB-430 platform. Therefore, we have used the *msp430-ram-usage* tool [15] that gives us an approximate value of the used RAM.

3.4 Energy Consumption

Energy is a very scarce resource in sensor nodes. Hence, it is crucial for the security architecture to retain a low energy overhead. The unique characteristics of sensor network applications make hardware-based energy measurement difficult [16]. In addition, the cost of a hardware-based mechanism for energy measurement is too high; the cost per hardware-unit is similar to the price of the sensor node [16].

In this project, we used *Energest* to evaluate the energy consumption of the cryptographic primitives. *Energest* is a software-based on-line energy estimation mechanism that estimates the energy consumption of a sensor node [17]. The mechanism runs directly on the sensor nodes and provides real-time estimates of the current energy consumption. *Energest* provides good estimates, but further study is needed to quantify the accuracy of the mechanism [17]. This tool maintains a table with entries for all components, such as CPU, radio transceiver, and LEDs. When a component starts running, a counter starts to measure the estimated energy consumption of this component. When the component is turned off, the timer is stopped.

4 Performance Characterization of Security Primitives under Contiki

4.1 Block Cipher Evaluation

The block ciphers that we have examined in this paper are briefly described below.

AES. Advanced Encryption Standard is a symmetric block cipher that can encrypt/decrypt data blocks of 128 bits, using cipher keys with lengths of 128, 192, and 256 bits. AES is based on the Rijndael algorithm, developed by Daemen and Rijmen in 1998 [18].

RC5. RC5 is a patented symmetric block cipher designed by Ronald L. Rivest of the MIT Computer Science and Artificial Intelligence Laboratory [19]. It is a parameterized block cipher with a variable block size, a variable key size, and a variable number of rounds.

Skipjack. Skipjack is an algorithm for encryption developed by the U.S. National Security Agency (NSA) and was declassified in 1998 [20]. It operates on data blocks of 64 bits with an 80-bit key.

Triple-DES. Triple-DES, also known as Triple Data Encryption Algorithm, is a variant of the Data Encryption Standard (DES) algorithm that executes the core DES algorithm three times [21]. It uses a 64-bit block size and a 168-bit key.

Twofish. Twofish is a symmetric block cipher with a block size of 128 bits and key sizes of 128, 192, and 256 bits [22]. It was designed by Schneier *et al.* for the AES contest.

XTEA. Extended Tiny Encryption Algorithm is a symmetric block cipher designed by Wheeler and Needham of the Cambridge Computer Laboratory [23]. XTEA is considered one of the fastest and most efficient algorithms. It operates on data blocks of 64 bits with a 128-bit key.

We focused on C language implementations of these ciphers since our objective is to integrate them into the Contiki operating system, which is designed in C. Due to the computational and memory limitations of sensor nodes, we used the most optimized, publicly available versions of each cipher.

In Figure 1, we show the time needed for encrypting and decrypting a single block of plaintext. Additionally, we show the time needed to carry out the key expansion process. For the context of these measurements it is important to emphasize the block size with which each cipher operates: XTEA, Skipjack, and Triple-DES operate with 64-bit blocks; in contrast, AES and Twofish operate with 128-bit blocks.

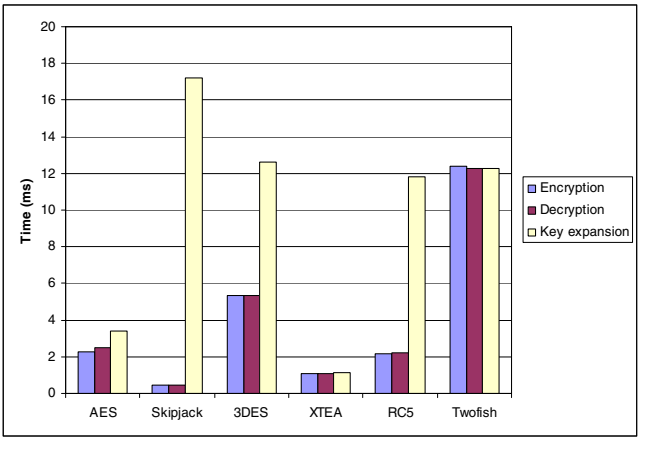


Fig. 1. Encryption, decryption, and key expansion time

From the figure above we can observe that Skipjack is the fastest cipher for encrypting/decrypting a single block of plaintext. However, it is the slowest cipher for the key expansion process. In our design, which we describe in the next section, we can have the key expansion process executed only when the node is initialized. Therefore, the time needed to do the key-setup process is not very significant. In order to simplify the analysis while taking into account the block size, we show the encryption throughput and the key length of each cipher in Table 1.

Table 1. Key length and throughput

Cipher	Key length	Throughput (Kbps)
AES	128	56.39
Skipjack	80	145.45
3DES	168	12.01
XTEA	128	59.26
RC5	128	29.49
Twofish	128	10.31

As mentioned, Skipjack has by far the best throughput. Nevertheless, if greater security is needed, we should choose a 128-bit key length cipher. In that case, AES and XTEA are the most suitable ciphers; both have similar encryption throughput. Their respective throughput is nearly three times smaller than the throughput of Skipjack.

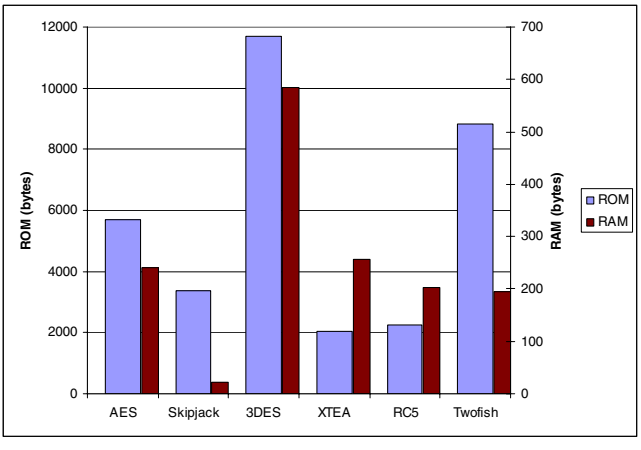


Fig. 2. ROM and RAM usage

In Figure 2 we show the memory requirements for the ciphers under consideration. We observe that Skipjack uses only 21 bytes of RAM, which is not a typical value. As we mentioned, the obtained RAM usage is an approximation. When analyzing the ROM usage, we see that XTEA uses the least ROM, followed by RC5 and Skipjack.

Energy is an extremely scarce resource in sensor nodes. Energy consumption of the cryptographic primitives must be taken into account when designing a security layer for WSNs. As expected, faster ciphers consume less energy. In Figure 3 we illustrate the estimated energy consumption of each block cipher.

These experimental observations show that Skipjack is the most efficient block cipher for our platform; in fact it is the best cipher with respect to throughput, RAM usage, and energy consumption. However, with the rapid advancement in computing, the 80-bit key in Skipjack is not completely secure. According to the claims of RSA

Security Labs, 80-bit keys would become *crackable* by 2010 [8]. Therefore, we believe that Skipjack should not be the default block cipher in security architectures.

The experimental results also show that XTEA is also a very efficient block cipher for WSNs. It achieves an acceptable throughput and it needs less ROM than any other cipher. Although it has a 128-bit key, a related-key differential attack can break 27 out of 64 rounds of XTEA [24].

After Skipjack and XTEA, AES achieves the best results in our experiments. It obtains a data throughput greater than radio rate, and reasonable memory usage and energy consumption. Additionally, at present time AES is the block cipher approved as a standard and recommended by NIST [25]. Moreover, Didla *et al.* demonstrated that AES could be highly optimized for WSNs [14]. Unfortunately, their optimized code is not in the public domain and is currently under a patent filing. Vitaletti and Palombizio also showed that AES can be effectively used in WSNs, and they have developed a module with AES for TinyOS [26].

Based on the results described above and taking into account the trade-off between security and resource consumption, we selected AES as the most appropriate block cipher for the WSNs under consideration.

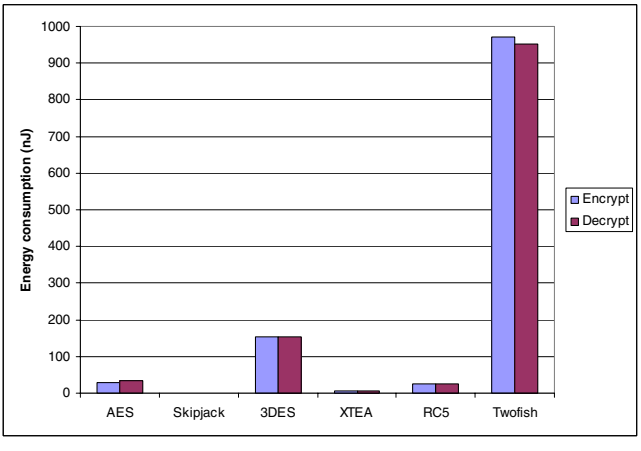


Fig. 3. Energy consumption for encrypting and decrypting one data block

4.2 CBC-CS Evaluation

Cipher Block Chaining–Ciphertext Stealing (CBC–CS) is a mode of operation proposed by NIST [27]. A mode of operation is an algorithm that features the use of a symmetric block cipher. A mode of operation requires an initialization vector (IV), which is a random block of data, to achieve the semantic security property. In Figure 4, we evaluate the mode CBC–CS with IVs of different sizes, using AES as the underlying block cipher.

From our observations, the overhead produced by padding the plaintext to the cipher block size is very important. Consequently, we should try to avoid encrypting

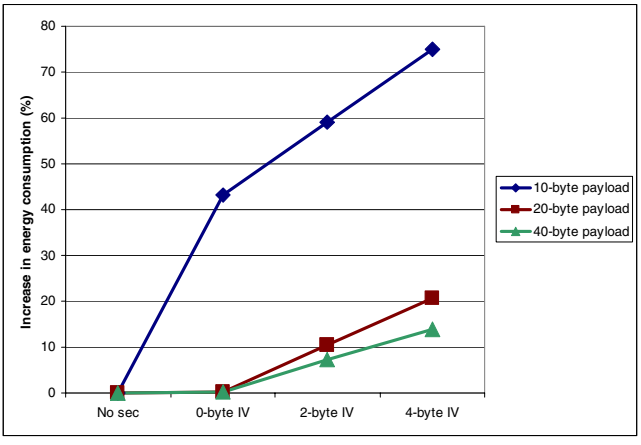


Fig. 4. The increase in energy consumption using CBC–CS–AES with different IVs

very short packets. In contrast, we observed that the power consumption for encrypting messages larger than the block size is nearly zero. In addition, sending an IV in each packet produces a significant increase in the energy consumption. The length of the IV should be defined depending on the security level required. In our opinion, a 2-byte IV makes a good balance between security and power consumption. In typical WSN scenarios, few packets per minute are sent, and thus the time to exhaust all the possible 2^{16} IVs is quite long [26].

4.3 CMAC Evaluation

Cipher-based Message Authentication Code (CMAC) is a cryptographic algorithm that provides assurance of the authenticity and, hence, the integrity of binary data [9]. In order to evaluate the impact of including a MAC in each message we have to define the length of the tag. Our experiments show that sending one byte has nearly the same power consumption as calling the AES encryption function six times. Thus, we have to limit the length of the MAC as much as possible. In TinySec [2] a 4-byte MAC is selected to provide authentication and integrity. The authors claim that a 4-byte MAC is appropriate for WSNs. In their paper they state: "*Adversaries can try to flood the channel with forgeries, but on a 19.2kb/s channel, one can only send 40 forgery attempts per second, so sending* 2^{31} *packets at this rate would take over 20 months.*" Furthermore, MiniSec and SenSec also use a 4-byte MAC. We also implement a 4-byte MAC in each message and we evaluate the influence on power consumption.

To determine the energy overhead of computing and adding a MAC in each message we performed several experiments with different data length. In Table 2 we show the obtained results.

Table 2. Energy consumption of sending packets with a MAC

Payload (bytes)	Mode	Energy (uJ)	Increase
10	Default	47.13	-
10	CMAC	53.43	13.37%
20	Default	82.48	-
20	CMAC	91.32	10.72%
40	Default	182.19	-
40	CMAC	195.75	7.44%

We observe that the energy cost of computing the MAC is negligible compared with the energy needed to send the MAC. The results demonstrate that the fixed overhead of sending each message (turning on the radio, sending the preamble and sync word) generally discourages short messages [2].

4.4 OCB Evaluation

Offset Codebook Mode (OCB) is a mode of operation that simultaneously provides confidentiality and authenticity [28]. When the application under consideration demands confidentiality and authenticity, OCB is claimed to be the most appropriate mode for WSNs in [3] and [8]. To evaluate the efficiency of OCB in our platform, we implemented this authenticated-encryption algorithm and compared it with the CBC–CS and CMAC modes.

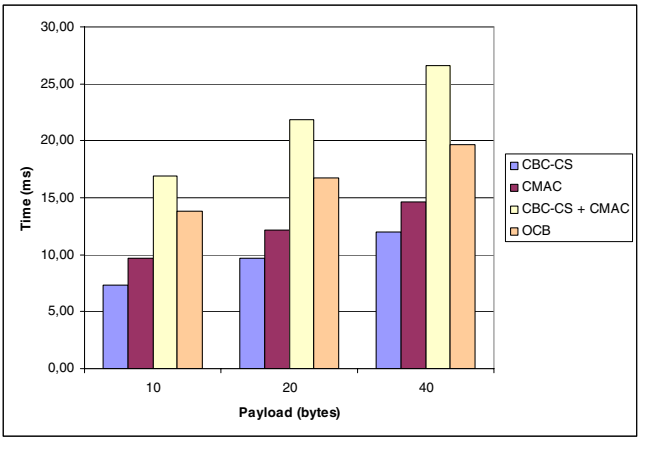


Fig. 5. Comparison of CBC–CS, CMAC, and OCB

As can be seen in Figure 5, OCB is faster than CBC–CS encryption combined with the CMAC. The results suggest that OCB is even more efficient with larger data lengths. Therefore, our analysis indicates that OCB is the best choice to select from when confidentiality and authenticity are required in the WSNs under consideration.

5 ContikiSec Design

Guided by the study, as presented in the previous section, we are now ready to put all the pieces together and describe the complete design of our secure network layer for wireless sensor networks under the Contiki operating system.

The Contiki protocol stack is divided in four layers: the physical layer, the data link layer, the communication layer, and the application layer. In WSNs, the typical traffic pattern is the many-to-one communication pattern. Because of that, in order to avoid routing packets injected by an adversary to waste energy and bandwidth, security should be implemented at the link layer [2]. Contiki provides three media access control protocols for our platform: X-MAC, LLP, and NULLMAC. Currently, we have implemented ContikiSec under NULLMAC, but it could be ported to run under the other two protocols. In Figure 6 we show the Contiki protocol stack including ContikiSec.

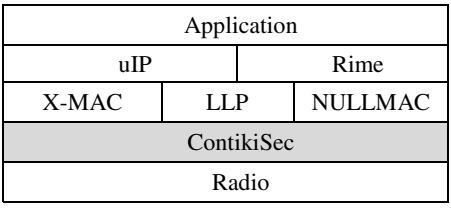


Fig. 6. The Contiki stack with ContikiSec

ContikiSec supports three security modes: confidentiality-only (ContikiSec-Enc), authentication-only (ContikiSec-Auth), and authentication with encryption (ContikiSec-AE). We believe that a configurable design is especially desirable for WSNs, as WSNs are expected to support many different application scenarios with different security requirements. ContikiSec offers the programmer the choice to select between three security levels depending on the needs of the application on hand. In Figure 7, we show the default Contiki packet format for the CC1020 radio and the packet format with different security configurations.

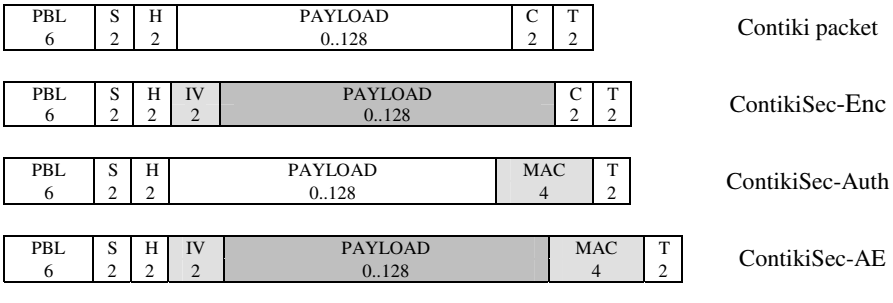


Fig. 7. ContikiSec packet formats compared to the Contiki packet format

As we can observe, ContikiSec-Enc includes a 2-byte initialization vector (IV) in each packet, producing a 2-byte overhead. The length of the IV has a great impact for security and energy consumption. Longer IVs are more secure, but the energy cost of sending them is too high for WSNs. In Section 4 we argue that a 2-byte IV makes the best balance between security and energy consumption. The IV is created using the Contiki library that generates random numbers. For the seed value to initialize the random generator, we used the node identifier. Hence each node initializes the random generator differently. The payload is encrypted using the CBC–CS mode of operation with AES as the underlying block cipher. Moreover, we assume that all the nodes in the network are provided with a single 128-bit key. The key-expansion process is performed when the node is initialized and then stored in the RAM. In addition, the CC1020 radio driver adds a 2-byte checksum field to each packet. Consequently, ContikiSec-Enc supports confidentiality and integrity.

ContikiSec-Auth is designed for applications where confidentiality is not critical, but where it is crucial to know the originator of each message. Contiki provides a 16-bit cyclic redundancy check (CRC) function to calculate a 2-byte checksum and achieve integrity. However, the checksum is designed to detect only accidental modifications of the data, whereas a MAC also detects intentional unauthorized modifications of the data. Hence, ContikiSec-Auth provides authentication and integrity by removing the checksum field from the packet and instead including a MAC. To generate a MAC, ContikiSec-Auth uses the CMAC algorithm benchmarked in Section 4, which is currently the mode for authentication recommended by NIST [9].

Finally, ContikiSec-AE provides the highest level of security, achieving confidentiality, authentication, and integrity. As we demonstrate in Section 4, the OCB mode is the most efficient algorithm when those properties are required. Consequently, ContikiSec-AE uses the OCB mode with AES as the underlying block cipher, using a single shared-key for encryption and authentication. ContikiSec-AE increases the packet length by 4 bytes. In Figure 8 we show the energy consumption of the three modes of ContikiSec.

As mentioned, and shown in Figure 4, the energy consumption for cryptographic operations is negligible compared to the energy needed for sending each extra byte. Hence, as the size of the payload is increased, the impact of sending extra bytes decreases.

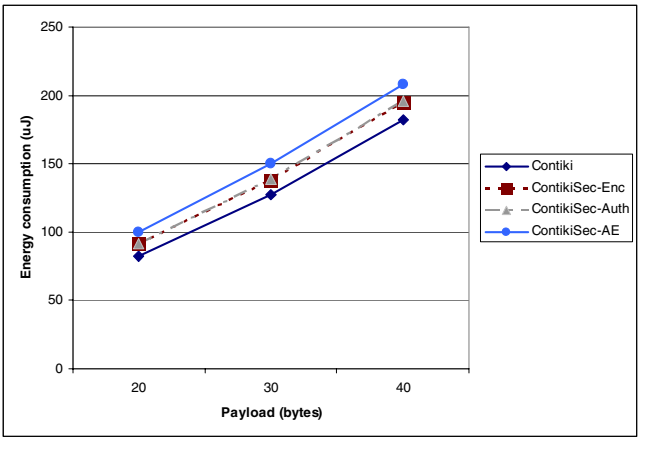


Fig. 8. Energy consumption of ContikiSec modes compared with the default Contiki configuration

Without security, the energy consumption ranges from 82 μJ to 182 μJ for packets with payloads of 20 to 40 bytes. Despite the fact that ContikiSec-Enc and ContikiSec-Auth have different computational costs, we observe the same energy consumption because both incur a 2-byte overhead. Finally, ContikiSec-AE is the mode with the highest energy requirement, consuming around 15% more than Contiki in default mode for data messages with payloads of 40 bytes.

6 Conclusion

We present ContikiSec, which is a secure network layer for wireless sensor networks under the Contiki operating system. We have designed ContikiSec as a complete and configurable solution, providing three security modes, starting from confidentiality and integrity, and increasing to confidentiality, authentication, and integrity. Our design was governed by a careful selection and performance analysis of existing security primitives. Our design tries to achieve low energy consumption without compromising security. Our evaluation was carried on the MSB-430 platform, a Modular Sensor Board hardware platform created by ScatterWeb.

In the future we plan to study the boundaries of using asymmetric keys in our framework and also examine the effect of compromised nodes in secure network layer architectures for wireless sensor networks.

References

1. Li, T., Wu, H., Wang, X., Bao, F.: SenSec Design. Technical Report-TR v1.1. InfoComm Security Department, Institute for Infocomm Research (2005)
2. Karlof, C., Sastry, N., Wagner, D.: TinySec: a Link Layer Security Architecture for Wireless Sensor Networks. In: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys 2004), Baltimore, pp. 162–175 (2004)
3. Luk, M., Mezzour, G., Perrig, A., Gligor, V.: MiniSec: a Secure Sensor Network Communication Architecture. In: Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN 2007), pp. 479–488. ACM, New York (2007)
4. Liu, A., Ning, P.: TinyECC: A Configurable Library for Elliptic Curve Cryptography in Wireless Sensor Networks. In: Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN 2008), SPOTS Track, pp. 245–256 (2008)
5. TinyOS, `http://www.tinyos.net`
6. Dunkels, A., Grönvall, B., Voigt, T.: Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors. In: Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, pp. 455–462. IEEE Computer Society, Washington (2004)
7. Perrig, A., Szewczyk, R., Wen, V., Culler, D., Tygar, J.D.: SPINS: Security Protocols for Sensor Networks. In: Proceedings of the 7th annual international conference on Mobile computing and networking, pp. 189–199. ACM, New York (2001)
8. Jinwala, D., Patel, D., Dasgupta, K.S.: Optimizing the Block Cipher and Modes of Operations Overhead at the Link Layer Security Framework in the Wireless Sensor Networks. In: Sekar, R., Pujari, A.K. (eds.) ICISS 2008. LNCS, vol. 5352, pp. 258–272. Springer, Heidelberg (2008)

9. National Institute of Standards and Technology, Computer Security Division. Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, Special Publication 800-38B (2005)
10. Chang, C., Nagel, D.J., Muftic, S.: Measurement of Energy Costs of Security in Wireless Sensor Nodes. In: Proceedings of 16th IEEE International Conference on Computer Communications and Networks (ICCCN 2007), pp. 95–102 (2007)
11. ScatterWeb, MSB-430datasheet, http://www.scatterweb.com/content/products/MSB_en.html
12. Texas Instruments, MSP430F1612 datasheet, http://focus.ti.com/mcu/docs/mcuprodoverview.tsp?sectionId=95&tabId=140&familyId=342
13. Chipcon, CC1020 datasheet, http://focus.ti.com/docs/prod/folders/print/cc1020.html
14. Optimizing AES for Embedded Devices and Wireless Sensor Networks. In: Presented at 4th International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom), Innsbruck, Austria (2008)
15. MSPGCC, http://mspgcc.sourceforge.net/
16. Jiang, X., Dutta, P., Culler, D., Stoica, I.: Micro Power Meter for Energy Monitoring of Wireless Sensor Networks at Scale. In: Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN 2007), pp. 185–195. ACM, New York (2007)
17. Dunkels, A., Österlind, F., Tsiftes, N., He, Z.: Software-based Sensor Node Energy Estimation. In: Proceedings of the 5th international conference on Embedded networked sensor systems (SenSys 2007), pp. 409–410. ACM, New York (2007)
18. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Springer, Heidelberg (2002)
19. Rivest, R.: The RC5 Encryption Algorithm. In: Proceedings of the 1994 Leuven Workshop on Fast Software Encryption, pp. 86–96. Springer, Heidelberg (1995)
20. National Institute of Standards and Technology, Computer Security Division. SKIPJACK and KEA Algorithm Specifications (1998)
21. National Institute of Standards and Technology, Computer Security Division. Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, Special Publication 800-67, Version 1.1 (2004)
22. Schneier, B., Kelsey, J., Whiting, D., Wagner, D., Hall, C., Ferguson, N.: The Twofish Encryption Algorithm. John Wiley & Sons, Chichester (1998)
23. Needham, R., Wheeler, D.: Tea extensions. Technical report, Computer Laboratory, University of Cambridge (1997)
24. Ko, Y., Hong, S., Lee, W., Lee, S., Lim, J.: Related Key Differential Attacks on 27 Rounds of XTEA and Full-Round of GOST. In: Roy, B., Meier, W. (eds.) FSE 2004. LNCS, vol. 3017, pp. 299–316. Springer, Heidelberg (2004)
25. National Institute of Standards and Technology, Computer Security Division, AES standard, http://csrc.nist.gov/archive/aes/index.html
26. Vitaletti, A., Palombizio, G.: Rijndael for Sensor Networks: Is Speed the Main Issue? In: Proceedings of the 2nd Workshop on Cryptography for Ad-hoc Networks (WCAN 2006). ENTCS, vol. 171, pp. 71–81 (2007)
27. National Institute of Standards and Technology, Computer Security Division. Proposal To Extend CBC Mode By Ciphertext Stealing (2007)
28. Rogaway, P., Bellare, M., Black, J.: OCB: A Block-Cipher Mode of Operation for Efficient Authenticated Encryption. In: ACM Transactions on Information and System Security (TISSEC), pp. 365–403. ACM, New York (2003)

A Mechanism for Identity Delegation at Authentication Level

Naveed Ahmed and Christian D. Jensen

Informatics and Mathematical Modelling
Technical University of Denmark
DK-2800 Lyngby
Denmark
nahm@kth.se, cdj@imm.dtu.dk

Abstract. Authentication and access control are normally considered as separate security concepts that have separate goals and are supported by separate security mechanisms. In most operating systems, however, access control is exclusively based on the identity of the requesting principal, e.g., an access control mechanism based on access control lists simply verifies that the authenticated identity of the requesting principal is on the list of authorized users.

In this paper we propose a human-to-human delegation mechanism for nomadic users, which exploits the amalgamation of authentication and access control in most operating systems, by delegating privileges at the identity level. The complexity of classic delegation models, especially if they strictly follow the principle of least privileges, often leads to a poor usability, which motivates a user to circumvent the default delegation mechanism. On the other hand, the identity delegation makes good use of trust relationships among users of a particular environment and offers the possibility of improved usability. Although identity delegation might violate the principle of least privileges, in practice it could increase the over all security of a nomadic environment where users need to delegate their duties frequently. The proposed mechanism is independent of the access control and the delegation event is only logged at the authentication level. Due to its improved usability, the motivation to share authentication tokens is reduced.

Keywords: Identity Delegation, Usability of Security, Nomadic User, Practical Security.

1 Introduction

Over the past few decades, we have experienced an exponential growth in computing technology and its pervasive use in our life has caused a shift in how we interact with computers. Earlier, in the age of mainframes and desktops, a user had to move physically to a computer for information or computation. When technology allowed manufacturing of lightweight devices along with a well connected wireless communication infrastructure, the paradigm of mobile

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 148–162, 2009.

computing emerged. Mobile computing allows people to move freely in an environment, while easily carrying their computing devices with them, e.g., mobile phone, laptops, PDA, etc. More recently, we have started embedding computing devices in work environments, all over the places where they might be required. This allows users to move freely in an environment without necessarily carrying their computing devices, as computers are already available at desired locations. We refer to this as nomadic use of computing and the corresponding nomadic users have the special usability requirements characterized by their frequent movement in a nomadic environment. We identify a nomadic user by having the following four properties: frequent movement; use of shareable devices; use of the devices that are part of the environment; and having a unique session.

Mutual collaboration of nomadic users is often required. This means that the support for user-friendly delegation is important in such environments. Nomadic users often need to work in each other's place, for instance in the nomadic environment of hospitals a senior doctor may need to delegate his duties to another doctor when a patient in critical condition arrives at the emergency unit. In such hectic environments, users are in hurry and often do not bother to use the default delegation mechanism and prefer to share their authentication tokens to delegate their duties, e.g., by password sharing, by handing over a pre-authenticated session, etc. This is mostly due to the short duration and frequent need for delegation, mutual trust of users, and the complex nature of existing delegation mechanisms that strictly follow the principle of least privileges – *Delegation should enable a delegator to grant only those rights required for performing a delegated task* [13,2].

We consider a Delegator as a person who delegates his authorizations to another person. The other person, who receives these authorizations, is called the Delegatee. Delegation mechanisms may be on a scale from fine grained, where each privilege is transferred individually, to coarse grained, where many privileges are transferred in one delegation. Fine grained delegation allows precise transfer of exactly those privileges that are necessary for a specific task, but the individual transfer of each privilege makes it cumbersome and error-prone, which gives users a strong incentive to circumvent the security mechanism. Coarse grained delegation, on the other hand, violates the principle of least privileges by transferring more privileges than strictly necessary, so both advantages and disadvantages must be considered in the context of the overall system security requirements when defining the granularity of delegation.

In the real world, there are often existing trust relationships among the users of a certain workplace and they play a significant role in the delegation, as in the earlier example of hospitals, the senior doctor trusts the other doctor for the professional honesty. In this type of situation, an easy delegation is much more important than the fine grained delegation of authorizations. Due to the notion of trust among co-workers, users often violate the system's security policy by deceiving the access control mechanism by using each other's authentication tokens.

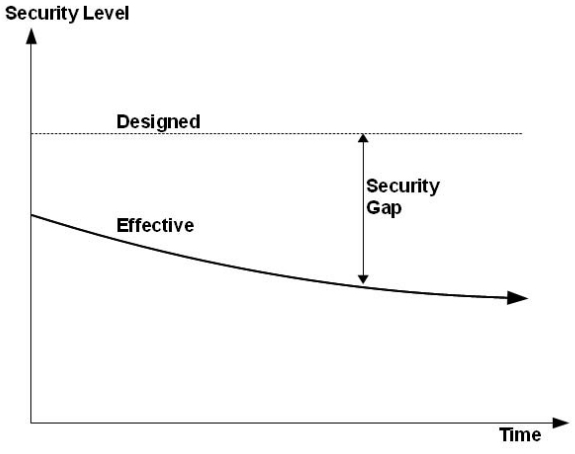


Fig. 1. Difference between Effective and Designed Security Levels

Delegation is necessarily regulated by a security policy, which defines the privileges of each user and their authority to delegate them. In a computer system, the security policy defines the intended level of security for the system, which we call the Designed Security Level. However, the security that we achieve in practice is often less than or at most equal to this intended level, we call this the Effective Security Level. Moreover, the effective security tends to decrease with time, mainly due to the discovery of new vulnerabilities in security mechanisms, advances in technology, improved techniques in cryptanalysis, usability issues, etc. This is also true for delegation mechanisms and is illustrated in Figure 1. In a nomadic environment, where users frequently delegate their duties, the poor usability of a complex delegation process provides a motivation to deceive the access control mechanism by sharing the authentication tokens, which contributes to the security gap indicated in the figure.

It is important to note, however, that the curve shown in the figure is for illustrative purposes only; the actual shape of the curve depends on the type of the system, the configuration and operation of the system, and parameters of the computational environment in which the system is deployed. For example, new security mechanisms, for which users are not yet trained or aware of its purpose, the curve of effective security could have the positive slopes in the start. Furthermore, finding good metrics for the accurate measure of these security levels is an open field of research [8].

In order to improve usability, we propose a delegation mechanism of a user's identity, which is at the authentication level. We assume that trust relationships exist among colleagues and co-workers which may justify violation of the principle of least privileges to a certain extent. In our proposed mechanism, the underlying access control mechanism is still deceived but we log delegation events at the authentication level and thus introduce accountability in a system. The improved usability reduces the burden of responsibility to correctly follow

the standard protocol for delegation and contributes to reduce the security gap indicated in Figure 1. The core idea behind the identity delegation is similar to Mercredi and Frey's work [12] but we propose an alternate and simpler architecture, which is derived from the usability factors of delegation in nomadic environments.

We have implemented the proposed delegation mechanism in a prototype system, which also uses RFID-based authentication. A user issues a simple command to delegate his identity. The delegatee can use his authentication token and the delegated identity to start the session of the delegator, by simply walking up to the system. The underlying access control mechanism does not distinguish between the delegator and the delegatee. In this way, the delegatee has all privileges of the session owner, however, the delegation event is logged for accountability. Due to this improved usability and simplicity, the motivation of sharing authentication tokens could be reduced.

In the next section, we describe part of the state of the art in delegation. In Section 3, we present our delegation mechanism in detail. In Section 4, we describe the prototype implementation of our proposed delegation mechanism. Section 5 provides some analysis results of the experiment, both in terms of security and usability. The final section presents our conclusion.

2 Related Work

The term, delegation, has many definitions in the literature, for instance Abadi et al. [1], Barka and Sandhu [4], Gasser and McDermott [6], Gladney [7], etc. However, in consistence with Zhang et al. [17], we consider it as a process in which one active entity in a system transfers its authority to another active entity in order for that entity to carry out a job on behalf of the first entity. Most of the time, the consequence of delegation is propagation and reassignment of the access rights. On a broader scale, we may distinguish between human-to-human, human-to-computer and computer-to-computer delegation. There are a number of delegation mechanisms for each of these, however, in this paper we focus on human-to-human delegation only.

Human-to-human delegation can be achieved in the two distinct domains of computer security. The most common way is to provide delegation support at the access control level. Gasser and McDermott [6] propose a technique for the delegation that not only provides the cryptographic assurances but also supports the authentication of a delegated system. They also claim the assurance for revocation even if the system is subsequently compromised. Varadharajan et al. [14] consider delegation in distributed systems as the problem of verification for a delegatee. They use a signature-based scheme for delegation with certain assumptions of trust relationships among the system entities. They also propose extensions to Kerberos mechanism to implement delegation, which otherwise does not support secure delegation in a straight forward way. Zhang et al. [17] propose a rule-based specification language and a rule-based framework for the role-based multi-step delegation. Although the Role Based Access

Control (RBAC) supports delegation at permission and role level but a formal framework for this purpose is proposed by Barka in his dissertation [5]. He has comprehensively characterized a delegation mechanism with seven attributes: Permanence, the distinction between permanent and temporary delegation; Monotonicity, the state of authorizations that a delegator possesses after the delegation; Totality, complete or partial delegation; Administration, the managing role in the delegation process; Level, single or multi step; Multiplicity, the number of people to whom the delegation is allowed at a given time; and Agreement, the authority of a delegatee to carry out a delegation. The delegation at permission level, although with very limited options, can also be accomplished in a standard UNIX and Linux access control model, where the owner of a resource can reassign its permissions to other group members. Jøsang et al. [9] propose a model for using a delegation network to manage authorizations.

The second domain, in which human-to-human delegation can be achieved, is user's authentication level. It may also be called user login or identity delegation and is quite independent of the underlying access control mechanism. The framework of a user login delegation is invented by Mercredi and Frey [12]. The primary objective of their framework is to address the problem in which a user allows another person to sign on at his behalf. In practice, this type of delegation is achieved by sharing authentication tokens or credentials, except for the biometric where sharing of an authentication token is not possible. Their framework also describes the delegation interface and the revocation in the form of generic flow charts. In UNIX and Linux, we can use sudo and su commands to invoke applications on the behalf of another user.

From the security point of view, the two domains of human-to-human delegation have an important difference. A delegation at access control level could follow the security policy of the system but an identity delegation cannot do so because the delegated identity enables the delegatee to use all those authorizations that are not required for the delegated job. However, on the other hand, identity delegation offers the possibility of a very user friendly interface, because all the necessary privileges are delegated in one full scoop along with the identity. This is not the case for the access control domain, where most of the time it is hard for a user to exactly figure out the privileges necessary for performing a particular job. Li and Wang [11] address this problem by bundling together the authorizations of a particular job in form of a unit, which they call Ability.

In the RBAC delegation model [5], where delegation is managed at the role and permission level, there is always a risk of under-delegation besides the complexity of finding out the relevant privileges for a delegated job. Under-delegation could be a serious problem in certain environments, e.g., safety critical systems, hospitals, etc. In UNIX and Unix-like systems, delegating by modifying file permissions or by changing group's memberships is an error-prone process as there is no special distinction for a delegated resource, besides it does not offer the explicit accountability of delegation. Once a delegation process is completed then,

at some point in future, it needs to be revoked, which might be non-trivial. Wang and Cao [15] categorize revocation schemes with four dimensions: dependency, resilience, propagation and dominance. Consequently, these four dimensions lead to sixteen different revocation schemes in a simplified RBAC model.

In nomadic use of computing, where users frequently delegate to one another, the complexity of delegation (and revocation) turns to poor usability and provides a strong motivation for users to bypass the secure way of delegation. As indicated by Bardram, et al. [3], users start sharing their authentication tokens, for instance, in the nomadic environment of hospitals, one should not assume that a doctor would bother to delegate only those access rights that are relevant to the certain patients in a ward, if he has to rush to the emergency unit. These usability issues of delegation contribute for the security gap between the designed level and the effective level of security.

The existing techniques of identity delegation using sudo or su commands of UNIX are very much implementation dependent. For instance, to use sudo command, a person first has to start his session with his own original identity. Moreover, it requires the delegated account password or explicit authorizations granted by the system administrator. The user login delegation described in the patent of Mercredi and Frey [12] assumes that the delegatee knows in advance about an inbound delegation and explicitly mention it while authenticating. It also does not cover the problem if two users have same password. Additionally, it is not focused to solve the usability problems that arise due to frequent delegations in a nomadic environment.

3 The Mechanism for Identity Delegation

The most of common way of ensuring computer security is access control mechanisms provided by operating systems such as UNIX, Linux, Windows, Mac OS, etc. An access control mechanism introduces a reference monitor for the objects in a system, which allows access to a particular object based on the subject's authorizations in the system security policy. However, in order to work securely, the access control mechanism must know the true identity of a subject. Since the subject's identity is validated by an authentication mechanism, the authentication is prerequisite for enforcing security with an access control mechanism.

A Validated Identity refers to the identity that an authentication mechanism concludes using one or more authentication techniques. An Effective Identity is the identity that is provided to the access control mechanism. The configuration shown in Figure 2 represents the delegation in an access control domain, with a user A as the delegatee and a user B as the delegator. As shown, the effective identity for the access control mechanism is the same as the validated identity of a delegatee A. The reference monitor grants all those access requests by the delegatee A that are either directly authorized to A or previously delegated to A by the delegator B.

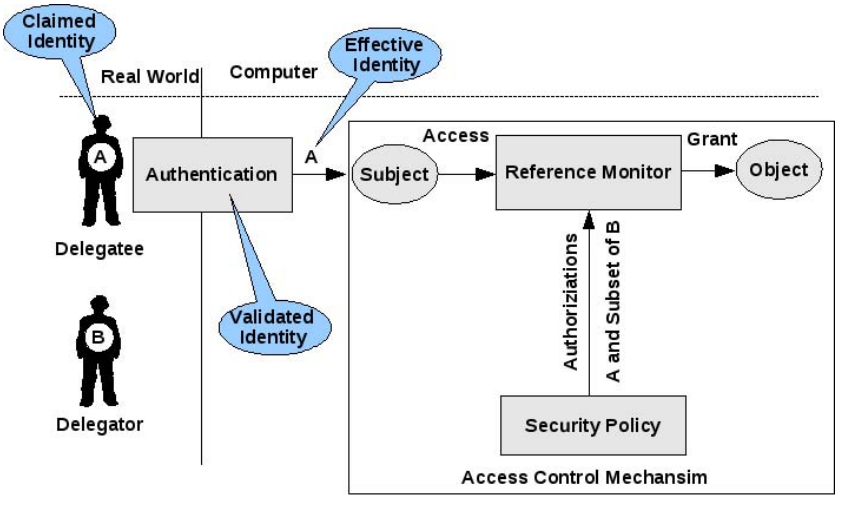


Fig. 2. Delegation in Access Control Domain

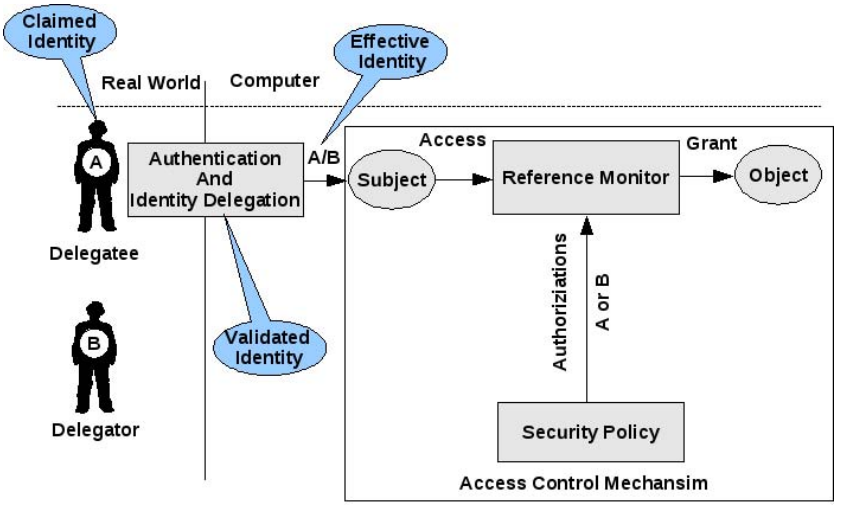


Fig. 3. Identity Delegation

On the other hand, we define identity delegation at the authentication level.

Definition 1. *If an authentication mechanism provides an effective identity different from the validated identity of the user then it is called identity delegation at the authentication level provided the owner of the effective identity has previously authorized the owner of the validated identity to use his identity.*

As shown in Figure 3, a user A can be recognized as a user B by an access control mechanism if B has previously delegated his identity to A. For example, let us consider the case of a classic login program where a person enters his

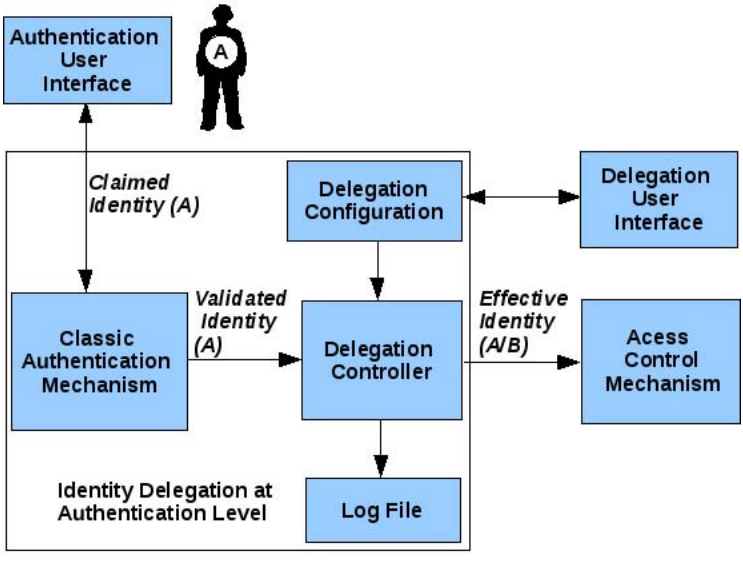


Fig. 4. Proposed Architecture for Identity Delegation

user name and the corresponding password. If the password matches in the local authentication database then the login program concludes a valid identity which it sends to the underlying access control mechanism of the operating system as the effective identity for the user. However, instead of passing on the validated identity, if the login program provides a different effective identity, we may call it identity delegation, provided the person who owns the effective identity has explicitly delegated this identity to the validated user, otherwise, it would be a security vulnerability known as impersonation.

The architecture of our proposed identity delegation is shown in Figure 4. As shown, the claimed identity of a user is evaluated by a classic authentication mechanism. On success, the claimed identity becomes the validated identity and then it passes through a delegation controller. The delegation controller maps the validated identity to the effective user identity based on the configuration in its delegation database. Finally, the effective user identity is provided to the access control mechanism. Since, in this case, the access control mechanism is unaware of any mapping, it enables us to make our identity delegation mechanism independent of any particular choice of the underlying access control mechanism. We provide a separate user interface in the user's session to modify the delegation configuration. Meanwhile, this delegation event is logged, in order to provide a level of accountability, which might be a justification for violating the principle of least privileges.

From the user perspective, the proposed mechanism is very simple as it only requires a user to decide whom he wants to delegate. The user does not need to worry about any permissions or roles. Moreover, the user does not need to modify the security policy, existing authorizations or refer back to the security

administrator. He simply issues a command from his session for the delegation and later on for the revocation. Due to this improved usability, we expect that the user would not be tempted to circumvent the delegation mechanism by sharing his authentication token. This could increase the effective level of system security in a nomadic environment where users need to delegate more often.

4 Identity Delegation Prototype

We have implemented the proposed architecture in form of a prototype which consists of a personal computer running Debian Linux. To further improve usability, we have augmented classic login based mechanism with RFID based authentication mechanism. Normally when no one is present, multiple sessions of users are suspended and the computer display is locked. When a user approaches, the relevant session is invoked instantly if RFID badge is validated. The relevant session could be the actual session of that particular user or it may be a delegated session. This selection depends on the current delegation configuration. However user can easily switch between the different delegated sessions by issuing a simple command.

The interaction of the different modules in the prototype is shown in Figure 5. The authentication Server, is the heart of the authentication mechanism as it grants or denies an authentication request using the identification which it receives from the RFID reader. Server is launched with the privileges of root user at startup, just before GNOME display manager(GDM) gets started. In GNOME based desktop, GDM manages sessions (X-Servers) of all users. Server in the prototype interacts with GDM, by sending commands in a standard protocol format through Clients, which run in each of the user's sessions. GDM controls the computer display and provides an abstraction for multiple sessions present on the computer. GDM also invokes the password based login program (GDMlogin), whenever a new session is started or reactivated. However, GDMlogin program is only a front end for interacting with a user for the user name and the password. Actual authentication is achieved by a pluggable authentication module associated with GDM. This association is described in a special file in the Linux configuration folder.

After the start of Server at startup, it takes the control of RFID reader through the serial port library. It communicates with RFID Reader, Clients, GDMlogin and DLG(Delegation) applications to achieve persistent(continuous) authentication and delegation by periodic scan of RFID tags. As shown in the figure, all these applications communicate with each other using network sockets. The use of network, for interprocess communication, enables us to port the authentication mechanism on an actual nomadic network where each terminal may run a remote session.

In typical use, a user carrying a valid RFID tag enters the interrogation field of RFID reader. RFID reader reports its identification to Server. Server checks this identification in the local database to find a match for a system user. If a match is found, Server maps the matched identification (validated identity) to a new

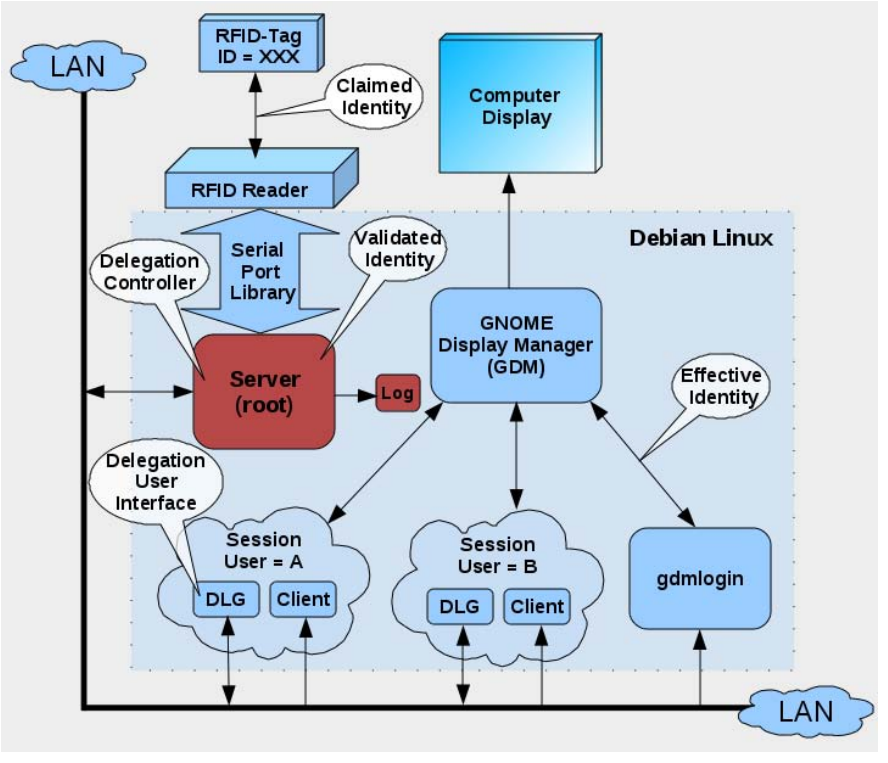


Fig. 5. Architecture of the Prototype

identification (effective identity) based on the current status of the delegation configuration. After this, it checks the availability of its session. In case relevant session is found, Server sends a command to the corresponding Client so that it can activate the session on the computer display. However, if the session is not found, Server creates a new session and waits for a predefined period of time to allow the new session to start. If a person is already authenticated and using his session then Server does not switch context even if other valid users come in the field of RFID reader. All these events are logged in a file for accountability.

For the delegation purpose, we have developed a console program which represents the delegation user interface of Figure 4. For instance if a user B wants to delegate his session to a user A, he issues following simple command.

```
>dlg set A
```

Similarly when A wants to switch to B's account, he uses following simple command.

```
>dlg switch B
```

And finally when B want to revoke the delegation, he uses following command.

```
>dlg reset B
```

Similarly there are other simple and easy to remember commands to display all inbound and outbound delegations, for group revocation, enabling context based delegations, etc. All these commands of delegation share a common motive, which is the simplicity of user interface as our aim is to make the delegation process to be user friendly so that people do not get motivated to circumvent it.

5 Evaluation

Similar to any delegation mechanism developed in the authentication domain, our mechanism violates the principle of least privileges [2]. However, this mechanism could be considered sufficiently secure in environments where there are pre-established trust relationships among users. For instance identity delegation may be considered secure if a doctor delegates his duties to his colleague but it might not be considered secure between anonymous parties. In real life, most delegations take place between people who do have certain level of trust as it is very unlikely that a doctor would allow a person, whom he does not know, to check his patients. Thus, we may consider the identity delegation to be practically secure for many collaborative and nomadic environments.

If the delegation is fine grained, like RBAC based delegation, then there is always a risk of under-delegation, i.e., the delegator does not delegate all the necessary permissions to perform a delegated job. This may cause the denial of service, which is very undesirable in some environments, such as in safety critical systems or in health care. In RBAC based delegation, one option to achieve delegation is by reassigning a set of permissions to the role of a delegatee, however, finding the relevant permissions for a particular job is not an easy task for large and complex systems. Moreover, by assigning these permissions to a delegatee role, all other users who are associated with that particular role get the delegated rights. Similarly, if the delegation is achieved by assigning the roles of a delegator to a delegatee then it would not only be a case of over-delegation but also the problem that the delegator has to figure out what roles, in complex hierarchy of RBAC, are necessary to perform a particular job. These types of problems are not present in our proposed delegation mechanism and the user interface is also very simple and easy to use.

Similar to the delegation process, revocation becomes non-trivial in some fine grained delegation mechanisms designed at access control level, which may cause failure to revoke all the delegated privileges. For example if a role receives the same delegated permissions from multiple delegators then revocation by one delegator should not actually revoke the delegatee's permissions. This implies that the revocation mechanisms must also keep track of all delegators at each step, for each permission and for each role, otherwise a denial of service may result. In our mechanism, the revocation is just a matter of issuing a simple command and since it is a part of the authentication it takes effect immediately causing the active session to be locked.

The classic access control models of UNIX and Unix-like systems may achieve delegation by changing file permissions and group memberships, but this also

implies that one should be either the owner of the resource or must have the super user privileges. A user is not essentially capable of delegating all privileges which he is authorized to use. In our identity delegation mechanism this problem is removed and delegation is only a user level decision.

There is no explicit notion of persistence (real-time responsiveness) in classic delegation models designed at access control level, such as Barka's delegation framework, or in Mercredi's user login delegation [12]. Since in our mechanism delegation is a part of authentication, thus a persistence authentication also implies a persistence delegation — delegation and revocation take effect immediately after the relevant command is issued.

In order to compare with the classic delegation in access control domain, we represent our proposed framework in a formal logic [16,1,10]. We start by introducing a few notions from this logic. When we write **A says S**, it implies that the entity A utters the statement S and believes in its truthfulness. We write $\mathbf{A} \Rightarrow \mathbf{B}$ to represent a kind of trust relationship in a way that whenever A makes a statement, B makes it too. Since the statement, made by a particular entity of the system, implies that the entity believes in it, thus the statement $\mathbf{A} \Rightarrow \mathbf{B}$ represents the trust of B on A. We write **A as R** to represent the entity A in a particular role R and consequently **A as R** has a subset of all authorizations that A normally possesses. $\mathbf{A'token}$ represents the role of A that possesses the authentication token of A. We write $(\mathbf{A} \mid \mathbf{B})$ **says S** to represent that A is quoting a statement S, of B. We write (**A for B**) **says S** to represent a quoted statement S and in addition, A is authorized to make such quoted statements on the behalf of B.

Each entity of the system has a set of authorizations, which it can transfer or delegate to other entities in the system using the following axioms.

$$If\ \mathbf{A\ says}\ (\mathbf{B} \Rightarrow \mathbf{A})\ then\ (\mathbf{B} \Rightarrow \mathbf{A}) \tag{1}$$

$$If\ \mathbf{A\ says}\ ((\mathbf{B} \mid \mathbf{A}) \Rightarrow (\mathbf{B\ for\ A}))\ then\ ((\mathbf{B} \mid \mathbf{A}) \Rightarrow (\mathbf{B\ for\ A})) \tag{2}$$

Statement (1) represents the transfer of authorities from A to B, while Statement (2) is the classic delegation of identity in access control domain as the identity of A is retained in all access requests. Now, let us consider Figure 4 and divide it in three system level entities: the user, A; the authentication and the delegation primitives enclosed in the rectangle of the figure, AD; and the access control mechanism, AC. We also assume that the only role of the authentication module AD is to provide the identity of a user. The default trust relationships between these entities are represented by the following statements.

$$\mathbf{AD} \Rightarrow \mathbf{AC} \tag{3}$$

$$\mathbf{A\ as\ A'token} \Rightarrow \mathbf{AD}\ and\ \mathbf{B\ as\ B'token} \Rightarrow \mathbf{AD} \tag{4}$$

In Statement (3), the access control mechanism trusts the authentication mechanism and hence the access control mechanism believes in the identity provided by

the authentication mechanism. In Statement (4), the authentication mechanism trusts a user with the role of possessing the correct authentication credentials in the form of a valid token that corresponds to the user in the authentication database. Statement (3) and (4) are enough to complete the trust chain from a user to the access control mechanism in order to invoke a user's session. For example, a user A with A'token can invoke A's session.

In our identity delegation mechanism, a user A may delegate his identity to another user B by sending a request to the authentication mechanism in the following form.

$$(\mathbf{A\ as\ A'token})\ \mathbf{says}\ ((\mathbf{B\ as\ B'token})|(\mathbf{A\ as\ A'token}))\Rightarrow \mathbf{A\ as\ A'token}) \tag{5}$$

An authenticated user A tells the authentication mechanism that if the authenticated user B quote a statement of the authenticated user A then it must be considered as the actual statement from the authenticated user A. Due to the formal logic axioms, the effect of Statement (5) is the following new trust relationship, which is in fact the identity delegation.

$$((\mathbf{B\ as\ B'token})|(\mathbf{A\ as\ A'token}))\ =>\ \mathbf{A\ as\ A'token} \tag{6}$$

Statement (6) represents that a request from B for the invocation of the session of A should be considered to be equivalent to a request directly made by A, which was in form of Statement (4). Note that the identity delegation at the authentication level represented by (6) is different from the transfer of authorities in (1) and the classic identity delegation of (2). In (1) the identity of the delegator A is completely lost and in (2) the identity of delegator is present in all requests made by the delegatee. In our mechanism (6) the identity of the delegator A is only present in the authentication and thus the access control mechanism does not distinguish between a delegator and a delegatee.

6 Conclusions

In nomadic environments, the usability of delegation is one of the decisive parameters in determining the effective level of system security. People in such environments mostly delegate to their co-workers and colleagues whom they trust. This fact could justify the identity delegation in nomadic environments despite its violation of the principle of least privileges. Since the identity delegation at authentication level can provide a simple user interface, it could increase the effective level of security by preventing a major motivation for users to circumvent complex delegation mechanisms.

We mitigate the poor usability associated with the classic delegation mechanisms, in order to reduce the gap between designed and effective level of security. The level of accountability in our mechanism helps to make the identity delegation usable for many nomadic environments by justifying the violation of the principle of least privileges. As our mechanism makes the delegation process very simple and user-friendly, many more people will use this secure way of

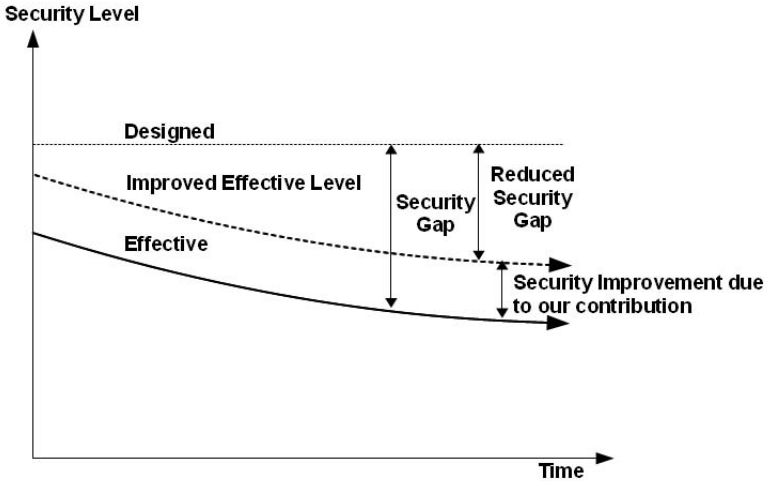


Fig. 6. Expected Contribution

delegation rather than deceive the access control mechanism to delegate their authorities. This improvement is indicated on Figure 6, which is adapted from Figure 1. In spite of the certain security improvement, however, we have not conducted an evaluation to determine exact quantitative values for the parameters or the exact shapes of curves shown in the figures. This is because, the actual values of the parameters vary with the type of system, the characteristics of users, underlying cryptographic primitives, protocols, etc. Therefore, it will be very hard to estimate them, but the trends indicated in the graphs are certainly true due to empirical observations and the analysis presented in the paper.

It is not always possible to achieve good usability and high security at the same time. That is why, security experts try to find a good trade-off between the two, under limited computational resources. Unfortunately, a good balance between usability and security is usually specific to a particular system under consideration. We also confess that a true analysis of a nomadic environment requires a cross-disciplinary approach, including anthropology and psychology, along with a couple of clinical and field trials, in order to truly discover the best usability options that may be incorporated in a delegation mechanism.

References

1. Abadi, M., Burrows, M., Lampson, B., Plotkin, G.: A calculus for access control in distributed systems. ACM Transactions on Programming Languages and Systems (TOPLAS) 15(4), 706–734 (1993)
2. Ahsant, M.: On-demand Restricted Delegation: A Framework for Dynamic, Context-Aware, Least-Privilege Delegation in Grids. PhD thesis, Kungliga Tekniska Högskolan (2009)

3. Bardram, J., Thomas, K., Nielsen, C.: Mobility in health care - reporting on our initial observations and pilot study. Technical Report CfPC 2003-PB-52, Center for Pervasive Computing (2003)
4. Barka, E., Sandhu, R.: A role-based delegation model and some extensions. In: Proceedings of 16th Annual Computer Security Application Conference, New Orleans, U.S.A. (December 2000)
5. Barka, E.S.: Framework for Role-Based Delegation Models. PhD thesis, George Mason University (2002)
6. Gasser, M., McDermott, E.: An architecture for practical delegation a distributed system. In: Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, California, U.S.A. (1990)
7. Gladney, H.M.: Access control for large collections. ACM Transactions on Information Systems 15(2), 154–194 (1997)
8. Gollmann, D.: Computer Security 2e. John Wiley and Sons, Chichester (2005)
9. Jøsang, A., Gollmann, D., Au, R.: A method for access authorisation through delegation networks. In: Proceedings of the 2006 Australasian workshops on Grid computing and e-research, pp. 165–174 (2006)
10. Lampson, B., Abadi, M., Burrows, M., Wobber, E.: Authentication in distributed systems: theory and practice. ACM Transactions on Computer Systems (TOCS) 10(4), 265–310 (1992)
11. Li, M., Wang, H.: ABDM: An extended flexible delegation model in RBAC. In: Proceedings of the 8th IEEE International Conference on Computer and Information Technology, Sydney, Australia, July 2008, pp. 390–395 (2008)
12. Mercredi, D., Frey, R.: User login delegation. United States Patent Application Publication, US 2004/0015702 A1 (January 2004)
13. Saltzer, J.H., Schroeder, M.D.: The Protection of information in computer systems. Proceedings of IEEE 63(9), 1278–1308 (1975)
14. Varadharajan, V., Allen, P., Black, S.: An analysis of the proxy problem in distributed systems. In: Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, California, U.S.A. (1991)
15. Wang, H., Cao, J.: Delegating revocations and authorizations. In: ter Hofstede, A.H.M., Benatallah, B., Paik, H.-Y. (eds.) BPM Workshops 2007. LNCS, vol. 4928, pp. 294–305. Springer, Heidelberg (2008)
16. Wobber, E., Abadi, M., Burrows, M., Lampson, B.: Authentication in the Taos operating system. ACM Transactions on Computer Systems (TOCS) 12(1), 3–32 (1994)
17. Zhang, L., Ahn, G.-J., Chu, B.-T.: A rule-based framework for role-based delegation and revocation. ACM Transactions on Information and System Security (TISSEC) 6(3), 404–441 (2003)

Introducing Sim-Based Security Tokens as Enabling Technology for Mobile Real-Time Services

Heiko Roßnagel[1] and Jan Muntermann[2]

[1] Fraunhofer Institute for Industrial Engineering (IAO),
Nobelstr. 12,
70569 Stuttgart, Germany
Heiko.Rossnagel@iao.fraunhofer.de
[2] Goethe-University Frankfurt, House of Finance,
Grüneburgplatz 1
60323 Frankfurt, Germany
muntermann@wiwi.uni-frankfurt.de

Abstract. We address inhibiting factors of mobile real-time services. Since research provides evidence that perceived risk is one major inhibiting factor for the adoption of mobile services, we explore current approaches that should safeguard existing security requirements. First, we develop a classification scheme of mobile real-time applications. We then empirically explore practical and security requirements for the different service classes. Our analysis indicates that current implementation concepts do not meet existent service requirements. Therefore, we present an alternative approach on the basis of SIM-based security tokens. The additional functionalities provided by these tokens can overcome functional or security constraints of existing implementations.

Keywords: Mobile Real-time Service, Security Requirements, SIM-based Security Tokens.

1 Introduction

Mobile services can provide significant value and benefits for users such as customers, employees and public servants and consequently for individuals, companies and the society in general [3]. In this paper, we explore the role of mobile real-time services which we define as services that use wireless communication networks and are provided without any significant time lag. While some of these services like news and sports result services have been highly successful, others such as mobile stock trading have not met market expectations yet. As these few empirical examples indicate, the motivation of this paper is derived from market research which currently finds high market penetration for some mobile services and low market penetration for others even though previous studies have predicted widespread diffusion also for the latter ones [49]. Since behavioural science-oriented research has shown that trust plays an important role for the adoption of certain mobile services [27] [28], we hypothesize that security requirements and inappropriate implementation approaches are key

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 163–178, 2009.

inhibitors for the adoption of these kinds of mobile real-time services. In order to corroborate or refuse this hypothesis we have developed a classification scheme that provides a methodological basis for classifying different mobile real-time services types. For the critical service types we have identified, we analyze their security requirements on the basis of exemplary cases. After exploring functional and security requirements of these services, we analyse if current approaches and existing implementations can meet these requirements. Based on shortcomings identified in this analysis, we present how the introduction of specific security token implementations can act as enabling technology for mobile real-time services by addressing both security and user requirements. The paper is structured as follows: The following Section 2 introduces the research domain of mobile real-time services. The development of a classification scheme of these services is given in section 3. Then, Section 4 provides a detailed analysis of the security requirements and possible security solutions for exemplary services being explored on the basis of this classification scheme. Section 5 examines the possible implementations of security modules to enable the exemplary services. Section 6 gives the conclusions of this study.

2 Mobile Real-Time Services

The specific value of mobile real-time services can be derived from user situations where ubiquity and reachability aspects play a crucial role. Such mobile real-time applications can be identified in time-critical arrangements or whenever spontaneous decisions and needs are supported [3]. Time-critical arrangements describe situations in which immediacy is worthwhile and where the near-ubiquitous connectivity aspect of mobile services comes into play. Such situations are triggered by external events the user might experience in the surrounding environment or of which the user is notified via push-based communication channels such as email or mobile messaging services [46]. There exist several corresponding service scenarios and cases such as event-triggered mobile decision support systems in the logistics and finance domain [34].

Spontaneous decisions and needs describe situations in which users want to request certain kind of information or to perform transactions while not being connected to a wired network [3]. Some successful examples comprise specific information services or ring tone purchases [47]. Unlike time-critical arrangements, most of these are not triggered by external events. In contrast, users do not rely on prompt information delivery or transactions to be performed quickly. Examples are general news services or standard bank transfers. Most of the mobile services in this field represent standard web-based service concepts that were shifted to the mobile service infrastructure [6]. Consequently, such services can scarcely benefit from increased ubiquity or reachability provided by the application of wireless communication channels [50].

To explore the interdependencies between functional and security requirements of mobile real-time services, we have first derived generic requirements from the mobile and real-time dimension, which are summarized in the following Table 1.

Table 1. Generic requirements of mobile real-time services

	Mobile	Real-time
Derived generic requirements	▪ (Near-)ubiquitous service availability ▪ Device interoperability	▪ Prompt data processing and delivery ▪ High service integration

Compared to classic e-commerce services, the relative advantage of mobile services is closely related to ubiquity aspects, i.e. users can benefit from increased availability of mobile services. Since there is a wide range of different mobile devices, successful mobile services are required to work across different device classes and mobile operating systems (device interoperability). The temporal facet mobile real-time services address demands for functionalities to ensure almost immediate user interaction. To provide such functionalities, data has to be processed and delivered promptly by the underlying service infrastructures. Furthermore, since many mobile services use different external data sources, high service integration is mandatory for providing real-time service capabilities. Whereas most of the current available mobile services aren't services in the sense described afore (such as the major mobile portals), they do not provide many benefits that could result from the functionalities mobile communication infrastructures provide. As a result, adoption of mobile services has not met expectations of early mobile commerce and business forecasts [40].

3 Classifying Mobile Real-Time Services

In the following, we further explore the promising domain of mobile real-time services. Therefore, we first conceptually define classes of mobile real-time services.

3.1 Introduction to Classification Techniques

Classification techniques play an important role in research since the classification of objects helps researchers to understand and analyze complex domains. The abilities to reduce complexity and to identify similarities and differences amongst objects are the major advantages provided by classifications [4]. One prominent classification scheme is Alter's taxonomy of decision support systems [2], which has often been applied by practitioners and used by researchers. We follow the common practice in social science research [4] and do first specify a classification scheme in section 3.2, which we then empirically explore in the following. Here, we focus on security aspects playing the central role in our analysis.

3.2 Classes of Mobile Real-Time Services

In order to explore the relevance of security for different classes of mobile real-time services, we define two dimensions in the following sections. This approach allows us to analyze interaction effects that might exist between different dimensions [45]. By combining these two dimensions, we present a sixfold classification scheme of mobile real-time services. In order to demonstrate its applicability, we present some empirical examples for each of classes identified in the next section 3.3.

3.2.1 Service Function

Using mobile real-time services can have two fundamental reasons: Users may desire information such as general news and routing information to be provided or may want to initiate a transaction [3]. The service function dimension is therefore first dichotomized as information service and transaction service. Furthermore, we identify two different types of information services by distinguishing between those services which are requested by the user (information pull service) and those which are proactively sent by an information service provider (notification push service). In contrast, transaction services are used in order to perform transactions that can be monetary or non-monetary.

3.2.2 Security Relevance

The requirements regarding relevance to security vary across different kinds of mobile services. Whereas some services demand for advanced security concepts (e.g. because of possible monetary losses), others don't. The security relevance dimension therefore classifies this aspect and is dichotomized as low and high security priority: Services with low security priority that demand for little or no security are usually not mission-critical in a sense that users don't rely on the service in terms of general functionality and security aspects. In contrast, there exist services that demand for high security priority. Such mobile services are critical in a sense that users demand for or rely on the supposed service functionality and security aspects. Otherwise, unwanted consequences can be expected.

Of course, a binary dimension to classify security priority is a rather simplistic view since the security characteristics and requirements of mobile services are complex. However, this simple classification scheme provides a conceptual basis to separate those services for which security aspects play a secondary role from those others for which it is at least of some importance. To analyze security requirements of specific service classes, several different (and maybe conflicting) security goals like integrity, confidentiality, availability and accountability have to be considered [35]. We will have a much more detailed look at such security requirements of specific mobile real-time applications in section 4.

3.3 Classification of Mobile Real-Time Services

We combine the two dimensions defined in the previous section and derive a classification scheme of mobile real-time services featuring 6 cells (Table 2).

In order to demonstrate the applicability of this scheme, we present a collection of exemplary cases:

- Information pull service with low security priority: These mobile real-time services e.g. comprise routing services that support users while traveling in unknown regions. In a typical scenario, such services are used by traveling salesmen on their way to customers.
- Information pull service with high security priority: Corporate intranet services are one example of this service type. When requesting time-critical information from a corporate backend system, the mobile worker must rely on the correctness of the information obtained. The same holds for personal information management (PIM) services

Table 2. Classification of mobile real-time services including some empirical cases

Security Priority	Information Services		Transaction Service
	Information Pull Service	Notification Push Service	
Low Priority	Routing services, Mobile games, News services, Weather forecasts, Education services/dictionaries	Flight schedule updates, Sport results, Recommendations (e.g. of pubs)	Entertainment Voting, Ring tone purchases
High Priority	Corporate intranet services, Financial information services, PIM services	Disaster management, PIM services	Stock trading Online auction bidding

- Notification push service with low security priority: In the sports domain, many fans continuously observe current scores of sporting events. A corresponding notification push service can proactively provide current changes of scores whereas usually only few demand for security exists.
- Notification push service with high security priority: Sending warning notifications to civilians in emergency situations to mitigate the impact of disasters is one example of this service type. So far, there exist only concepts or prototypes of emergency management systems utilizing mobile services. In the addressed scenarios, people receive time-critical information via mobile push services such as mobile text messages.
- Transaction services with low security priority: This service type is characterized in that faulty or not executed transactions do not lead to high monetary losses. Exemplary services comprise entertainment voting (e.g. used by the worldwide successful chart-shows) or purchases of ring tones.
- Transaction services with high security priority: Here, potential monetary losses or other unwished consequences support the demand for security. Mobile stock trading and online auction bidding represent exemplary representatives of this service type.

The created classification scheme and the service type examples provide the conceptual and empirical basis for a detailed security analysis that we present in the next section.

4 Empirical Evidence

Having defined our problem domain we are now able to focus on those mobile real-time services for which security aspects play a major role. In this section, we will take a close look at sample applications for the different types of service functions. We will start by describing a security relevant information pull service along with its

security requirements. This will be followed by similar descriptions of a notification push service and a transaction service. Our analyses are based on exemplary mobile real-time services in order to achieve better clarity of our arguments and to provide empirical evidence for its applicability. For our analysis, we used the confidentiality, integrity, and availability (CIA) triangle that forms the fundamental basis of IT security [48] [26]. However, since availability has already been identified as a generic requirement for all mobile real-time services, we will not further elaborate its obvious necessity. Instead, we added the security goal of accountability [35] to our analysis.

4.1 Information Pull Services

4.1.1 Example: Secure Corporate Intranet Services

There is a need of corporations to provide their mobile workforce with access to the corporate intranet, in order to supply them with accurate and actual information [21]. For example, the external sales force can be supported with real-time information regarding product availability, prices and details. Figure 1 shows such an application scenario where a salesperson has mobile real-time access to corporate intranet services.

Since the mobile users are located outside the security domain of the company it is very important to make sure that access to this information service is appropriately secured [43]. Therefore, access to the information service should only be granted to clients that have been securely identified and authenticated [12]. Furthermore, the confidentiality of the communication between the mobile client and the corporate backend must be protected [19]. In order to avoid spoofing attacks [15] the mobile user should be able to verify the authenticity of the information service.

In addition, the integrity of the information must be protected during the transmission. Table 3 summarizes the security requirements of the described pull information service.

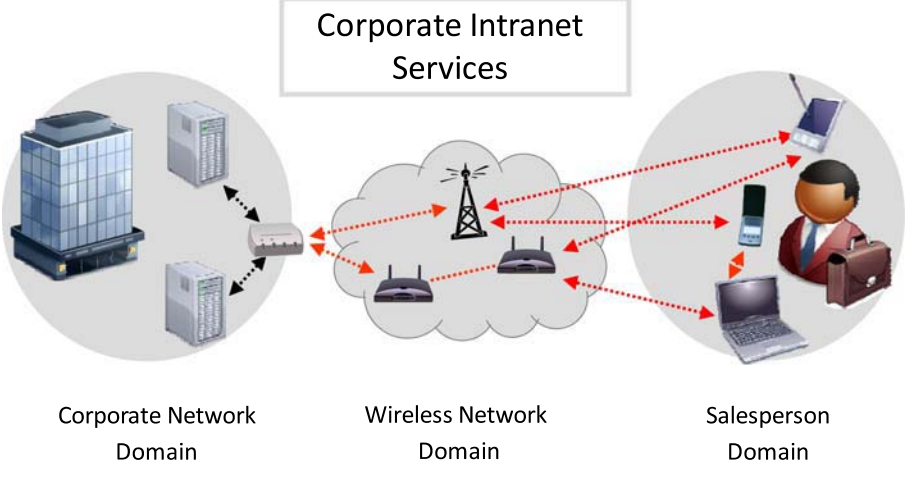


Fig. 1. Mobile Real-time Corporate Intranet Services

Table 3. Security requirements for mobile real-time corporate intranet services

Security Goal	Requirement
Accountability	▪ Access to the information service should only be granted to clients that have been securely identified and authenticated. ▪ The mobile user should be able to check the authenticity of the information service.
Confidentiality	▪ The confidentiality of the communication between the client and the corporate backend must be protected.
Integrity	▪ The integrity of the information must be protected during transmission.

4.1.2 Implementation Alternatives and Their Implications

Most of the requirements can be met by using Secure Socket Layer (SSL) and SSL-server certificates respectively. To achieve secure client authentication one or more of three fundamental approaches could be used [31]: something the user knows (password); something the user has (token) and something the user is (biometric).

Current implementations usually use passwords or external security token to allow this functionality [25] [12]. However, users tend to either choose weak passwords [51] [8], or choose related passwords for several or even all accounts [1], which makes the authentication system vulnerable to cross-service attacks [24]. External tokens on the other hand are expensive and stored on extra hardware that has to be connected to the device, needs to be carried around and can be lost easily [12]. This can conflict with the functional requirements of device interoperability, prompt data processing and ubiquitous service availability (in case of a token lost) and clearly does not fulfill the functional requirement of high service integration. Also, these tokens are usually proprietary solutions for one particular service. This can lead to situations in which a security token is used for corporate intranet access, while the company email will be received using BlackBerry [37] or a similar service, which could induce new vulnerabilities [43]. The use of a two factor authentication with a service independent security module present on the client device in combination with an authentication secret or biometric identification seems desirable. Depending on the security policies of the corporation, different ways to implement such a security module are possible (see section 5).

4.2 Notification Push Services

4.2.1 Example: Mobile Emergency Management

One example for a security relevant push notification service can be found in the field of emergency management. The goal of an emergency management system is to enable disaster forces to manage disaster events, including detection and analysis of incidents [9]. Persons in charge should be supported to prepare evacuations, control and support disaster forces and to locate victims [9]. Since mobile communication infrastructures offer standardized wireless communication services in almost all countries [17] and allow a fast diffusion of information, they provide a new, previously unused technological basis for saving human lives in emergencies. In case a disaster event occurs, the disaster manager sends out warnings to the affected areas by cell

broadcast to ensure prompt warnings of potential victims [16]. Disasters caused by human failure or deliberate attacks usually do not have any leading signs, such as Chernobyl explosion or the September 11th attacks. Also people usually have problems to recognize leading signs of natural disasters and the possible magnitude of damages [14]. One tragic example is the 2004 Tsunami. In such scenarios, promptly and secure notification and information supply is especially desirable [23] [13]. A text message received on the mobile phone is much easier to interpret as for example the different signals used by sirens, which is a major advantage. Therefore, the mobile notification service can offer warnings that are sufficiently understandable and action-oriented and can offer confirmation, which is the ideal form of a warning [7]. Therefore, the Dutch Government for example has decided to deploy nationwide a emergency management system based on mobile communications [10]. Figure 2 shows such an application scenario where a person located in a danger zone receives an alerting notification.

Along with the new opportunities of fine-grained emergency management, new possibilities of abuse such as fake disaster warnings do emerge [29]. Emergency

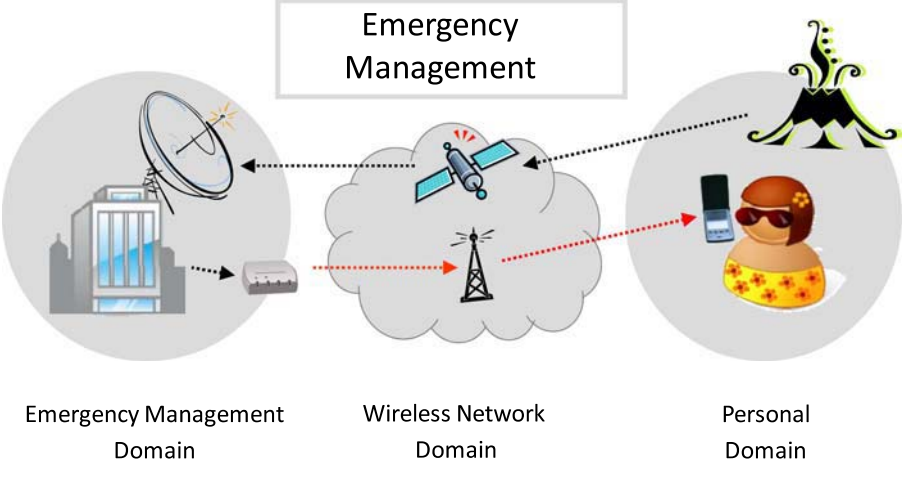


Fig. 2. Mobile emergency management

Table 4. Security requirements for mobile emergency warning

Security Goal	Requirement
Accountability:	The notification service has to ensure that the user can determine that the notification is from an authorized disaster management authority.
Confidentiality:	-/-
Integrity:	The notification service has to ensure that the user can determine that the integrity of the notification message has not been violated.

managers have to ensure that misleading or contradictory information is not disseminated [38]. Therefore, the notification service should provide means to ensure the authenticity and integrity of the warning message [41].

Confidentiality of warning messages is not desired, since the purpose of the warning is to reach as much people located in the disaster zone as possible.

4.2.2 Implementation Alternatives and Their Implications

Fulfillment of these security requirements would necessitate modifications of the systems currently in use. Neither SMS nor Cell Broadcast Service (CBS), which offers the most suitable solution for warning civilians [22], offer the possibility to automatically check the authenticity or integrity of the received warning information. Therefore, both communication methods are vulnerable to faked or altered messages. Consequently, the necessary security techniques have to be implemented on top of the existing infrastructures. Roßnagel and Scherner [41] propose the usage of electronic signatures to ensure the authenticity and integrity of the sent information. However, this would require a security module on the mobile device to check the validity of these signatures. This approach raises major implementation issues. Security modules need to be distributed to users well before a possible disaster occurs. Also the functional requirement of device interoperability has to be considered for such an implementation. Therefore, the presence of a device independent security module working on all major devices forms a necessary precondition of such a notification service.

4.3 Transaction Services

4.3.1 Example: Mobile Stock Trading

Trading opportunities that arise from mobile financial information and stock trading services can provide significant value to private investors as they can be informed permanently about relevant market events by mobile push services [32]. However, solely increasing the level of information will not lead to prompt investment success. Investors must be empowered to perform necessary transactions in time or they will not be able to take advantage of the better information situation [33]. Figure 3 illustrates such an application scenario in which an investor is supported by a mobile investment transaction service.

Security issues play a critical role for the adoption of mobile banking services. A study on mobile banking security shows that nearly 85% of potential customers are worried about online security of financial transactions, including credit card purchases [18]. Therefore, it is quite obvious that security is essential also for mobile stock trading services. Naturally, access to a transaction service should only be granted to clients that have been securely identified and authenticated. In order to avoid spoofing attacks, the investor should be able to verify the authenticity of the information service. In addition, the communication between the investor and the online broker should be confidential. The transaction service should also ensure that transactions are properly authorized by the investor and that the integrity of the intended transaction will be preserved. Table 5 summarizes these security requirements.

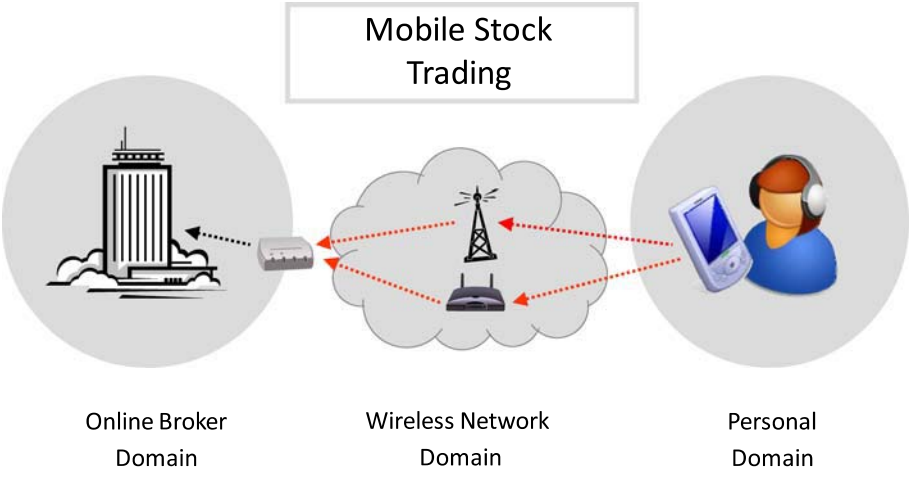


Fig. 3. Mobile stock trading

Table 5. Security requirements for mobile stock trading

Security Goal	Requirement
Accountability	▪ Access to the transaction service should only be granted to clients that have been securely identified and authenticated. ▪ The mobile user should be able to check the authenticity of the transaction service. ▪ Transactions have to be securely authorized by the investor.
Confidentiality	▪ The confidentiality of the communication between the client and the transaction service must be protected.
Integrity	▪ The integrity of the transaction should be ensured.

4.3.2 Implementation Alternatives and their Implications

Most of the mobile brokerage services currently available and analyzed by the academic community are implemented like traditional online brokerage services [5]. In these traditional mobile banking scenarios, a personal identification number (PIN) is used to authenticate the investor. To authorize a transaction, the investor has to enter a transaction number (TAN). Confidentiality is provided by SSL and the authenticity of the transaction service can be verified by checking the server certificate. Despite being widely used, this implementation concept has several shortcomings. While being pressed for time, it is very inconvenient for the investor to enter a transaction number for each transaction as he or she has to carry a list of TANs at all times in order to be able to react to observed market events. The application of key code lists has been identified as one of the key inhibitors for the adoption of mobile banking services [30]. Also, this increases the risk of potential theft or loss of the TAN list. Furthermore, the usage of PIN/TAN authentication has been shown to be vulnerable against phishing attacks [15]. There is also no way of ensuring the integrity of the ordered transaction. The main problem is that the TAN is not bound to the intended transaction, but rather a mechanism for entity authentication for a single transaction session

[11]. Some of these shortcomings explain low adoption rates of current mobile banking and brokerage services [44]. According to Muntermann and Roßnagel [33], most of these problems could be addressed by the presence of a security module. The investor could perform stock orders using electronic signatures, which will be created by the security module. In order to sign a transaction, the investor has to enter the signature PIN. This signature PIN does not change, eliminating the need to carry around a TAN list. The transaction server verifies the signature and completes the transaction if the signature is valid. By signing the transaction request the investor ensures the integrity of this request. It also guarantees that only the investor could have requested this transaction enabling accountability

4.4 Summary of Results

In addition to the basic needs of mobile real-time services as described in Section 2, additional requirements for security relevant services have been identified based on the empirical evidence in the previous subsections. Table 6 summarizes these requirements.

Table 6. Summary of security requirements

Security Goal	Information Services		Transaction Service
	Corporate Intranet Services	Emergency Management	Stock Trading
Accountability	+	+	+
Confidentiality	+	-	+
Integrity	+	+	+

As stated before, the security goal of availability has not been included in our security analysis, because its necessity derives from the generic requirement stated already in section 2. While some requirements can be met by functionalities of current mobile communication technology infrastructures, other requirements could only be fulfilled by the presence of a security module. Table 7 provides an overview of the requirements that can be met with and without the use of a security module.

Table 7. Fulfilment of requirements with and without security modules

Security Goal	Information Services				Transaction Service	
	Corporate Intranet Services		Emergency Management		Stock Trading	
	No SM	SM	No SM	SM	No SM	SM
Accountability	-	+	-[1]	+	-	+
Confidentiality	+[2]	+	(-)[3]	(+)[3]	+[2]	+
Integrity	+[2]	+	-	+	-	+

[1] SMS and CBS provide no means for verifying the authenticity or integrity of the notification.
[2] Can be accomplished by using SSL/TLS.
[3] Confidentiality is not required for this service type.

As illustrated in Table 7, the presence of a security module, that performs the necessary cryptographic algorithms, would make it possible to fulfill all of the requirements presented in this overview.

5 Security Module as Enabling Technology

Based on the empirical evidence described in section 4 we conclude that the presence of a security module and the resulting fulfillment of additional requirements would enable new mobile real-time services. Furthermore, the security of present mobile services could be improved, which would mitigate existing adoption barriers. There are several different ways to implement such a security module. It could either be a software component running on the mobile phone, some external security token that can be connected to the mobile phone or the SIM-card that is already present in a mobile phone. Using software that is running on the mobile phone or an external security token would, however, have a significant negative effect on the interoperability of the services. Since mobile subscribers use a great variety of different mobile devices, services should be as compatible as possible to most devices on the market. Using a software solution would force service providers (or mobile operators) to implement security modules for a variety of different platforms. Similar problems occur when external security tokens are used. They either must be connected to the mobile phone by some sort of compatible interface or they require the user to enter security codes, which would reduce the convenience of the service. Also, if the security modules are provided by the service providers, the security module is usually only useable with this one particular service. If the user wants to use several security-relevant services, he or she will most likely be confronted with several different security modules. Using one security module for all security-relevant mobile real-time services would be much more desirable. One such alternative to eliminate service interoperability issues could be realized by using the SIM-card as security module, which is provided by the mobile operator. The desired device interoperability could also be achieved by implementing the envisioned services using the SIM Application Toolkit (SIM AT) which ensures compatibility to almost all mobile phones [20]. However, this would require a SIM-card with a cryptographic co-processor to perform the necessary operations. The technology for such SIM cards exists but has not gained much market penetration so far. For example the WiTness project [36] sponsored by the European Union developed such a SIM card that is capable of creating RSA signatures [39]. Exchanging the SIM cards will induce additional costs for the mobile operator. These costs have to be compensated by an increase in mobile communication traffic caused by the new applications, which are based on a security module. In [42] the profitability of such an exchange has been researched. The author concludes that - given a promising service use case – the investment into such a technology can be profitable for mobile operators.

6 Conclusion

The adoption of certain mobile service types has not met market expectations so far. Especially in the area of mobile real-time services, which are used in time-critical

arrangements or whenever spontaneous decisions and needs are supported, several services that would be of significant value to mobile users and could be highly profitable for service providers have not been realized or have not achieved high market penetration yet. While services that require only minimal security like entertainment voting or news push information services have been successful, security relevant services like stock trading or emergency notification are not being offered or scarcely in use so far. We hypothesized that the reason for this development is that the security requirements of such services can not be met appropriately by current service implementations. In order to explore this hypothesis, we have developed a classification scheme of mobile real-time services that provides a methodological basis for classifying different mobile real-time services types from a security-perspective, and to achieve a clear distinction of services. Exploring the different service types, it becomes clear that those mobile real-time service types for which security is of major importance have either not yet achieved high market penetration or business needs have led to proprietary implementations that pose further problems. For these service types we provided empiric examples which we further explored in detail. Based on this empirical evidence we identified the need for a security module to be present on the mobile device as a prerequisite for the implementation of such services. Of our exemplary services only one – corporate intranet access - has been employed widely on the basis of external hardware tokens. This of course is not ideal from a usability perspective and these solutions are proprietary and not interoperable. The existence of such solutions, however, demonstrates the immense market demand for this service. Aside from hardware tokens, software or crypto-enabled SIM cards could also be used as security modules. Using crypto-enabled SIM cards as security modules offers the advantages of service and device interoperability, while at the same time providing the security of hardware tokens. However, issuing such SIM cards will induce additionally costs for mobile operators. These costs have to be compensated by an increase in mobile communication traffic caused by the new services, which are based on a security module. Our classification could help to identify such mobile real-time service that will be enabled by the existence of a security module. If enough of those services can be identified or one very compelling use case exists, this could lead to a strong incentive for mobile operators to invest into the necessary technology in the future.

References

1. Adams, A., Sasse, M.A., Lunt, P.: Making Passwords Secure and Usable. In: Proceedings of HCI on People and Computers XII, pp. 1–19. Springer, Bristol (1997)
2. Alter, S.: A Taxonomy of Decision Support Systems. Sloan Management Review 19(1), 39–56 (1977)
3. Anckar, B., D'Incau, D.: Value-Added Services in Mobile Commerce: An Analytical Framework and Empirical Findings from a National Consumer Survey. In: Dolk, D.R. (ed.) Proceedings of the 35th Hawaii International Conference on System Sciences. IEEE Computer Society, Los Alamitos (2002)
4. Bailey, K.D.: Typologies and Taxonomies: An Introduction to Classification Techniques. Sage, Thousand Oaks (1994)

5. Barnes, S.J., Corbitt, B.: Mobile Banking: Concept and Potential. International Journal of Mobile Communications 1(3), 273–288 (2003)
6. Barnes, S.J.: The mobile commerce value chain: analysis and future developements. International Journal of Information Management (22), 91–108 (2002)
7. Botterell, A., Addams-Moring, R.: Public Warning in the Networked Age: Open Standards to the rescue? Communications of the ACM 50(3), 59–60 (2007)
8. Brown, B.J., Callis, K.: Computer Password Choice and Personality Traits Among College Students, Southeast Missouri State University, Cape Girardeau, Missouri (2004)
9. Carver, L., Turoff, M.: Human Computer Interaction: The Human and Computer as a Team in Emergency Management Information Systems. Communications of the ACM 50(3), 33–38 (2007)
10. Cisco Systems: Unique mobile communications solution transforms disaster and emergency response capabilities in the Netherlands, http://www.cisco.com/web/strategy/docs/gov/Cisco_SVS.pdf (accessed 2008-02-18)
11. Claessens, J., Dem, V., de Cock, D., Preneel, B., Vandewalle, J.: On the Security of Today's Online Electronic Banking Systems. Computers & Security 21(3), 257–269 (2002)
12. Clarke, N.L., Furnell, S.M.: Advanced user authentication for mobile devices. Computers & Security 26(2), 109–119 (2007)
13. Currion, P., De Silva, C., Van De Walle, B.: Open Source Software for Disaster Management: Evaluating how the Sahana disaster information system coordinates disparate institutional and technical resources in the wake of the Indian Ocean tsunami. Communications of the ACM 50(3), 61–65 (2007)
14. Faulkner, B.: Towards a Framework for Tourism Disaster Management. Tourism Management 22, 135–147 (2001)
15. Felten, E.W., Balfanz, D., Dean, D., Wallach, D.S.: Web Spoofing: An Internet Con Game. In: Proceedings of the National Information Systems Security Conference, Baltimore, Maryland, pp. 95–103 (1997)
16. Fritsch, L., Scherner, T.: A Multilaterally Secure, Privacy-Friendly Location-based Service for Disaster Management and Civil Protection. In: Proceedings of the AICED/ICN 2005, pp. 1130–1137. Springer, Heidelberg (2005)
17. Ghani, R.: The Future of Wireless Banking, http://www-106.ibm.com/developerworks/library/wi-banking/?article=wir (accessed 26.2.2008)
18. Ghosh, A.K., Swaminatha, T.M.: Software Security and Privacy Risks in Mobile E-Commerce: Examining the risks in wireless computing that will likely influence the emerging m-commerce market. Communications of the ACM 44(2), 51–57 (2001)
19. GSMworld: GSM Operators, Coverage Maps and Roaming Information, http://www.gsmworld.com/roaming/gsminfo/index.shtml (accessed 2008-02-07)
20. Guthery, S.B., Cronin, M.J.: Mobile Application Development with SMS and the SIM Toolkit. McGraw-Hill, New York (2002)
21. Haller, J., Robinson, P., Walter, T., Kilian-Kehr, R.: Framework and architecture for secure mobile business applications. In: Gritzalis, D., De Capitani di Vimercati, S., Samarati, P., Katsikas, S.K. (eds.) Security and Privacy in the Age of Uncertainty, Proceedings of the IFIP TC11 International Conference on Information Security (SEC 2003), pp. 413–416. Kluwer, Athens (2002)
22. Harris, I.: LS - Use of SMS and CBS for Emergencies, ETSI, Tokyo, March 9-11 (2005), http://www.3gpp.org/ftp/tsg_t/TSG_T/TSGT_27/Docs/PDF/TP-050051.pdf
23. Horan, T.A., Schooley, B.L.: Time-Critical Information Services. Communications of the ACM 50(3), 73–78 (2007)

24. Ives, B., Walsh, K.R., Schneider, H.: The Domino Effect of Password Reuse. Communications of the ACM 47(4), 75–78 (2004)
25. Karjoth, G.: Access Control with IBM Tivoli Access Manager. ACM Transactions on Information and System Security 6(2), 232–257 (2003)
26. Kesh, S., Ratnasingam, P.: A Knowledge Architecture for IT Security. Communications of the ACM 50(7), 103–108 (2007)
27. Kleijnen, M., Ruyter, K.d., Wetzels, M.: Consumer Adoption of Wireless Services: Discovering the Rules, While Playing the Game. Journal of Interactive Marketing 18(2), 51–61 (2004)
28. Luarn, P., Lin, H.: Toward an understanding of the behavioral intention to use mobile banking. Computers in Human Behavior 21, 873–891 (2005)
29. Manoj, B.S., Hubenko Baker, A.: Communication Challenges in Emergency Response. Communications of the ACM 50(3), 51–53 (2007)
30. Mattila, M.: Factors Affecting the Adoption of Mobile Banking Services. Journal of Internet Banking and Commerce 8(1) (2003)
31. Mayes, K.M.K., Piper, F.: Smart Card based authentication: Any Future? Computers & Security 24, 188–191 (2005)
32. Muntermann, J., Janssen, L.: Assessing Customers' Value of Mobile Financial Information Services: Empirical-Based Measures. In: Proceedings of the Twenty-Sixth International Conference on Information Systems (ICIS 2005), Las Vegas, NV, USA, pp. 617–628 (2005)
33. Muntermann, J., Roßnagel, H.: Security Issues and Capabilities of Mobile Brokerage Services and Infrastructures. Journal of Information System Security 2(1), 27–43 (2006)
34. Ngai, E.W., Gunasekaran, A.: A review for mobile commerce research and applications. Decision Support Systems 43(1), 3–15 (2007)
35. Pfitzmann, A.: Multilateral Security: Enabling Technologies and Their Evaluation. In: Müller, G. (ed.) ETRICS 2006. LNCS, vol. 3995, pp. 1–13. Springer, Heidelberg (2006)
36. Project Wireless Trust for Mobile Business: Deliverable D4: SIM Application Hosting: Detailed Description of the Concept, Munich, Germany (2002)
37. Research in Motion: BlackBerry, http://www.blackberry.com/ (accessed 7.2.2008)
38. Riley, J., Meadows, J.: The Role of Information in Disaster Planning: A Case Study Approach. Disaster Prevention and Management 6(5), 349–355 (1997)
39. Rivest, R.L., Shamir, A., Adleman, L.: A Method for Obtaining Digital Signatures and Public Key Cryptosystems. Communications of the ACM 21(2), 120–126 (1978)
40. Berger, R.: Dial M for Mobile: Key Success Factors in the Wireless World, New York (2000)
41. Roßnagel, H., Scherner, T.: Secure Mobile Notifications of Civilians in Case of a Disaster. In: Leitold, H., Markatos, E.P. (eds.) CMS 2006. LNCS, vol. 4237, pp. 33–42. Springer, Heidelberg (2006)
42. Roßnagel, H.: Mobile Qualifizierte Elektronische Signaturen: Analyse der Hemmnisfaktoren und Gestaltungsvorschläge zur Einführung, Gabler, Wiesbaden (2009)
43. Schulz, E.: Mobile Computing: The next Pandora's Box. Computers & Security 26(3), 187 (2007)
44. Shen, S., Pittet, S., Milanesi, C., Ingelbrecht, N., Hart, T.J., Desai, K., Nguyen, T.H., Basso, M.: Overview of Consumer Mobile Applications, Gartner Research, Stamford, CT, USA (2006)
45. Stinchcombe, A.L.: Constructing Social Theories. Univ. of Chicago Press, Chicago (1987)
46. Sun, J.: Information Requirement Elicitation in Mobile Commerce: Using an interactive Approach to Facilitate Information Searching for Mobile Users. Communications of the ACM 46(12), 45–47 (2003)

47. Sun, S., Su, C., Ju, T.: A study of Consumer Value-added Services in Mobile Commerce: on Domestic Cellular Phone Companies in Taiwan, China. In: Li, Q., Liang, T. (eds.) Proceedings of the 7th International Conference on Electronic Commerce. ACM Press, New York (2005)
48. Swanson, M.: Security Self Assessment Guide for Information Technology Systems: Special Publication 800-26, National Institute for Standards and Technology, U.S. Government Printing Office, Washington, DC (2001)
49. Van Veen, N., Reitsma, R., Carini, A.: The European Mobile Landscape 2006: European Consumer Technology Adoption Study, Forrester Research, Cambridge, MA, USA, June 12 (2006)
50. Venkatesh, V., Ramesh, V., Massey, A.P.: Understanding Usability in Mobile Commerce: Ramifications for wireless Design: E<>M. Communications of the ACM 46(12), 53–56 (2003)
51. Yan, J., Blackwell, A., Anderson, R., Grant, A.: Password Memorability and Security: Empirical Results. IEEE Security Privacy 2(5), 25–31 (2004)

Towards True Random Number Generation in Mobile Environments

Jan Bouda, Jan Krhovjak, Vashek Matyas, and Petr Svenda

Faculty of Informatics, Masaryk University
Botanicka 68a, 602 00 Brno, Czech Republic

Abstract. In our paper, we analyze possibilities to generate true random data in mobile devices such as mobile phones or pocket computers. We show how to extract arguably true random data with a probability distribution $\epsilon = 2^{-64}$ close to the uniform distribution in the trace distance. To postprocess the random data acquired from the camera we use a randomness extractor based on the Carter-Wegman universal$_2$ families of hashing functions. We generate the data at the bit rate approximatively 36 bits per second – we used such a low bit rate only to allow statistical testing at a reasonable level of confidence.

Keywords: min-entropy, random number generator, randomness extractor.

1 Introduction

A good generator of random numbers for cryptographic purposes must guarantee good statistical properties and unpredictability of its output. The whole generation process is not a trivial task and it must utilize nondeterministic sources of randomness together with high-quality digital postprocessing. Without physical randomness the output and all derived cryptographic key material could be (easily) predictable. Digital postprocessing by randomness extractors or pseudorandom number generators is necessary to improve statistical properties of generated data and to offer protection against limited malicious manipulation of the source of randomness.

Many documented flaws in implementations of random or pseudorandom number generators exist – e.g., in an early version of Netscape SSL [GW96], OpenSSL FIPS Object Module [OS07], Java 2 ME [KMC07], or Sun's MIDP Reference Implementation of SSL [SMH05]. They have a critical impact to the security of many common services. For example, a recently published flaw (resulting in predictability of the generator) in the Debian OpenSSL caused that SSH keys, SSL certificates, DNSSEC keys, OpenVPN keys, and DH/DSA session keys generated in the past two years should be considered compromised [DS08].

In our paper, we study options for true random number generation in mobile devices that contain no special purpose hardware designed especially for this task. We examine and describe randomness sources available to the application programmer in current mobile phones and other mobile devices, e.g., the

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 179–189, 2009.

Nokia N73 with the Symbian OS that we used for our tests. The main criteria for true random number generation in our setting are provable security against active adversaries and sufficient bit rate of the source.

The paper is structured in the following way: In Section 2 we discuss possible sources of randomness we can use and in Section 3 we describe how to process the acquired (biased) random data to obtain (almost) uniformly distributed randomness. In Section 4 we review randomness extractors, choose extractors suitable for our purposes and describe one particular extractor used in our tests. The tests performed to verify quality of randomness that we input to the extractor and obtain from the extractor are described in Section 5. Finally, in Section 6 we discuss possible ways to improve the bit rate and computational efficiency of the extractor.

2 Sources of Randomness

We performed several practical experiments on two Nokia N73 smartphones (with the Symbian OS) and two similar PDA phones E-Ten X500 and E-Ten M700 (with the Windows Mobile OS) [KSM07]. The goal of our experiments was to identify (and assess the quality of) sources of randomness in these mobile devices and to estimate the amount of randomness (min-entropy) in these sources. We used two identical Nokia N73 devices since we wanted to verify the correctness of our results or to detect unexpected behavior of the smartphone – in a case when one device suffers, e.g., by some manufacturing defect.

Due to the API restrictions (especially in the Symbian OS), we were forced to drop sources of randomness like the battery level, signal strength or GPS position as measurements over these sources do not provide output (at the API level) with a sufficient precision (e.g., battery and signal values are available in the form of an integer between 0 and 10) or frequency (e.g., external GPS provides only one measurement per second). On the contrary, microphone and digital camera perform a high-rate sampling of physical sources, yielding high volumes of data. Since we can never guarantee the quality of a physical source, our analysis is concentrated on the microphone and the camera noise that arises, e.g., in the CCD/CMOS chip or the A/D converter, and is always present in the output data.

Unfortunately, digitalized microphone input is slightly correlated, and captured camera frames contain, due to low-level postprocessing and optical sensor technology (such as row-dependent readouts), several systematic defects. These defects result in a decrease of the entropy of probability distribution supplied by these sources. Moreover, all external sources of randomness can be influenced (gamma rays, temperature, etc.) and protection against such attacks requires advanced defence techniques. Changes of temperature have an impact to the noise level that slightly influences the probability distribution, but we have not discovered any regularities that could make the output more easily predictable. On the contrary, direct illumination of CCD/CMOS chip can result in generation of highly biased values – overexposed pixels yield always the value 255. Detailed discussion of these issues can be found in [KSM07].

Digital camera provides higher data rate than the microphone and this implies also a higher entropy (and min-entropy) yield in the same time period. We used "view finding" (resolution 180×240, a standard function of Nokia N73 smartphone) rather then a full-size image to get the raw image data without almost any postprocessing. Such raw data were captured at temperatures of 8, 20 and 45 degrees Celsius to observe the influence of environment temperature on the camera noise arising from chip. Unless explicitly mentioned all results in this paper refer to data obtained from the Nokia N73 digital camera at 8 degrees Celsius – i.e., data with the smallest amount of noise (and min-entropy).

In order to use our sources of randomness for cryptographical purposes, we need to obtain a sequence of random bits distributed according to the uniform distribution. The problem to solve is that our device not only outputs (in general) a biased distribution, but this distribution in not known in advance since it might be influenced by an adversary. To postprocess our data and to get an output with some provable guarantees of the probability distribution we have to use some randomness extractor.

3 Processing Randomness

The straightforward way to construct the postprocessing software would be to estimate the probability distribution of camera output first ($180 \cdot 240 \cdot 8 \cdot 3 = 1\,036\,800 \approx 2^{20}$ sample points), or at least the min-entropy of the distribution, and then design a suitable randomness extractor. The obvious obstacle to this method is the size of sample space that prevents reasonable statistical testing (the number of samples would be practically unreachable).

We will limit the amount of data used from each frame to 4 bits only to allow statistical testing. In order to use more information from each frame at least to increase expected min-entropy we would not take 4 raw bits directly obtained from the camera, but rather as a function of a number of bits.

To extract random bits from a CCD/CMOS chip we adopted the following general approach (see Figure 1):

1. Acquire a picture (frame) from a CCD/CMOS chip of the camera (when the lens is closed).
2. Apply a function f (see below) that distills few (in our case four) random bits from the input frame.
3. Repeat step 1. and 2. until a sufficiently long output sequence is accumulated. Then input this sequence to the randomness extractor to obtain the final random sequence.

The role of function f is to preprocess the random data in the way that regardless of the (limited) actions of the adversary the probability distribution of sequences of outputs of function f has with high probability min-entropy sufficient for successful randomness extraction (for chosen extractor). The main problem when designing function f is the limited knowledge of specification of the CCD/CMOS chip and the postprocessing done by the camera. Moreover, these details change

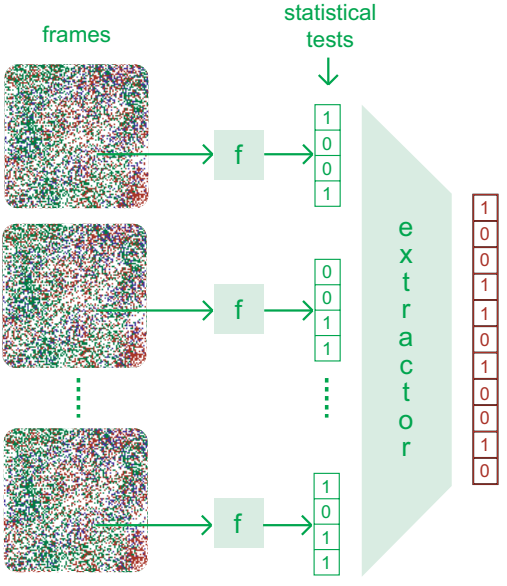


Fig. 1. Scheme of the Data Postprocessing

with both model and producer of the mobile device. We use the function f to separate the extractor from these technical assumptions.

Following the aforementioned motivation we designed the function f to be an XOR (parity) of least significant bits (LSBs) of selected (colors of) pixels of the CCD/CMOS chip. The choice of LSBs has the obvious motivation that this bit is hard to predict and to flip this bit a very accurate (physical) attack is needed[1].

We make one important assumption in our analysis – that bits obtained from different frames are independent. To assure this, we cyclically change pixels used as inputs for the function f. Also, we limit the sampling frequency to 12 frames per second.

Let us concentrate on the noise arising from the CCD/CMOS chip in the Nokia N73 smartphone. The generator we considered takes 12 pictures per second from the view finding (resolution 180×240 function of the cell phone), while the lens of the camera is shut and therefore the influence of the output by external lights is minimized. Randomness contained in pictures comes mainly from the heat inducing electrical charge in cells of the CCD/CMOS chip. As we mentioned before, we use a function f that extracts 4 bits from each picture. We proceed with sampling sufficient number of 4-bit sequences to estimate the probability distribution our source delivers with reasonable confidence. We run the same tests in various external conditions that may influence the source (cold, heat – 8, 20 and 45 degrees Celsius) and obtain limitations on probability distributions an adversary can achieve. More details on performed tests are presented in Section 5.

[1] As a further improvement we later discuss XORing all bits of particular (colors of) pixel instead of LSBs only.

More precisely, we used 180×240 pixels from each frame. Function f logically divides pixels in one frame to 300 squares of 12×12 pixels and uses exactly one pixel from each square. These pixels are selected deterministically in such a way that the neighbors are always from different row and column. The main motivation is that we get a pixels from different locations of CCD/CMOS chip and we mitigate possible dependencies of pixels from successive frames.

Finally, we extract only four bits from these 15×20 squares/pixels. Firstly, we apply XOR function to all pixels (and colors) in a column. Then we transform the resulting 20 bytes (1 byte per column) into 4 bytes by XORing together several different non-neighbor columns: $(1, 5, 9, 13, 17)$, $(2, 6, 10, 14, 18)$, $(3, 7, 11, 15, 19)$, and $(4, 8, 12, 16, 20)$. At the end we apply XOR also to all bits in each byte, or alternatively extract only the LSB from each byte. The output is in both cases exactly four bits per frame – to improve inter-frame pixel independence we took the resulting pixels in order $1, 3, 2, 4$, respectively. For statistical properties of the output see Section 5.

Note that a successful attack scenario defeating such preprocessing technique would require an attacker capable of repeatedly flipping or predicting all bits used for construction of all resulting four bits per frame. Aside from aforementioned need of very accurate physical attack to CCD/CMOS chip, flipping one particular bit in image captured from "view finding" will be very tricky, since algorithms used in mobile devices for downgrading the CCD/CMOS image resolution to image with a "view finding" resolution are kept secret.

4 Randomness Extractor

In this section we review the theory of randomness extractors, choose extractors suitable for our purposes and describe one particular extractor used in our implementation and corresponding tests.

4.1 Definition of Randomness Extractor

The randomness extractor is a function $e : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ that takes n bit random (biased) input, d bit auxiliary uniformly distributed random input and outputs m bit sequence that should be distributed according to the (almost) uniform probability distribution (see below). The processed data obtained from the camera are used as the first parameter, while the second parameter requires true random uniformly distributed data. This means that such data must be pregenerated, and it can be, e.g., stored in the memory and refreshed via network after a fixed number of extractor iterations (meanwhile the output of the extractor is used). This is an obvious drawback, but such an extractor allows us to process a wide range of input probability distributions, namely any probability distribution with sufficiently high (pre-chosen) amount of randomness.

Definition 4.1. *The* **min-entropy** *of a probability distribution* $(p_1, \ldots, p_{2^n})$ *is*

$$H_\infty(p_1, \ldots, p_{2^n}) = \min_{i=1,\ldots,2^n} -\log p_i = -\log \max_{i=1,\ldots,2^n} p_i. \quad (1)$$

Definition 4.2. *Let X and Y be random variables defined on the same sample space $\mathcal{S}$ with probability distributions p_X and p_Y, respectively. We say that X and Y are* ϵ**-close in the trace (or** L_1**)** *distance iff*

$$\frac{1}{2}\sum_{a\in\mathcal{S}}|p_X(a)-p_Y(a)| = \max_{A\subseteq\mathcal{S}}|P(X\in A)-P(Y\in A)| \leq \epsilon. \tag{2}$$

Definition 4.3. *The function*

$$e : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m \tag{3}$$

is a (non-deterministic) (k,ϵ) **randomness extractor** *if for every input distribution X on $\{0,1\}^n$ with $H_\infty(X) \geq k$ and uniform probability distribution on $\{0,1\}^d$ the probability distribution of the output is ϵ-close in the trace distance to the uniform probability distribution on m bit strings.*

The motivation why to use the min-entropy in the randomness extractor definition is straightforward – it is impossible to extract $d+k$ random bits using an extractor with d bit auxiliary input provided the min-entropy of the input is $k-1$ or less. On the other hand, this bound can be approached asymptotically [Sha02].

The other possibility is to use a deterministic extractor $e : \{0,1\}^n \to \{0,1\}^m$ requiring no uniformly distributed input. Unfortunately, possibilities of such a source are strongly restricted – it partitions all inputs into fixed subsets, each of which is mapped to one fixed output bit sequence. Therefore, this extractor gives unbiased output only as long as the probability of each such subset is unchanged. As an extreme example, given a fixed randomness extractor with n-bit input and single bit output, there is always a probability distribution on n-bit strings with min-entropy $n-1$, such that the output of the extractor on this distribution is deterministic, i.e. the distribution $(0,1)$ [Sha02].

It may be impossible to explicitly construct one suitable deterministic extractor, however, non-deterministic extractor with random, but public and fixed auxiliary input can do very well [BST03] in some situations. For more details on randomness extractors see the survey paper [Sha02].

Finally, in this paper we use extractors that guarantee uniform distribution of concatenated extractor output and auxiliary input. Hence, the useful almost-uniform output of such an extractor has $d+m$ bits.

Definition 4.4. *The function*

$$e : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m \tag{4}$$

is a (k,ϵ) **strong randomness extractor** *if for every input distribution X on $\{0,1\}^n$ with $H_\infty(X) \geq k$ and uniform probability distribution Y on $\{0,1\}^d$ the joint probability distribution $\big(Y, e(X,Y)\big)$ is ϵ-close in the trace distance to the uniform probability distribution on $m+d$ bit strings.*

4.2 Extractors Based on Carter-Wegman Universal Hashing

To fight a general adversary we decided to use a (non-deterministic) randomness extractor, i.e., extractor requiring true randomness as an auxiliary input. This allows us to concentrate only on the analysis of the min-entropy of our randomness source. Due to a high min-entropy of our source (see Section 5) we can choose freely from a wide variety of extractors.

First parameter of extractor we should consider is the length d of the auxiliary random input. In our case this should be considered only in the context of the randomness efficiency, i.e., we should compare the length of the output sequence with the amount of randomness in the input, i.e., the length of the output should be approaching $d + k$. Possibly limiting factor of the initial sequence can be memory of the mobile device, but essentially all devices with a camera already have memory of size (at least) few megabytes (few hundreds for contemporary mobile phones) and therefore we are not limited by a reasonably large auxiliary input that has to be stored (we consider approx. 1000 bits for our purposes).

Since the absolute value of the true random input is not limiting and we have a high min-entropy, we can concentrate on the computational efficiency of our extractor. Computing power of mobile devices is still limited and intensive calculations increase battery exhaustion and thus the computational efficiency is of critical importance. The amount of true randomness is limited, and, therefore, we would like to reuse it or use the output of our extractor instead. A straightforward choice from this point of view are extractors based on Carter-Wegman universal classes of hash functions.

Definition 4.5 ([CW79]). *A class* $H = \{h|h : \{0,1\}^n \to \{0,1\}^m\}$ *of hash functions is* **universal**$_2$ *iff for every* $x, y \in \{0,1\}^n$ *there are exactly* $|H|/2^m$ *functions* h *such that* $h(x) = h(y)$.

Theorem 4.6 ([SZ99, IZ89, IL89]). *Let* X *be a random variable defined on the sample space* $\mathcal{S} = \{0,1\}^n$ *with probability distribution* p *having min-entropy* $H_\infty(p) \geq k$, $H = \{h|h : \{0,1\}^n \to \{0,1\}^{k-2e}\}$ *be a universal*$_2$ *class of hash functions. Let* $x \in_R \{0,1\}^n$ *be randomly chosen from* $\{0,1\}^n$ *according to* p *and* h *be randomly and uniformly chosen from* H. *Then the distribution of* $(h, h(x))$ *is* 2^{-e} *close to the uniform distribution in the trace distance, i.e. application of a function randomly chosen from* H *is a* $k, 2^{-e}$ *strong randomness extractor.*

As follows from the previous theorem, with each application of the extractor we loose $2e$ bits in contrast to the theoretic achievable rate. Note that this is fixed and independent of n, k and $|H|$ and, therefore, by using longer input sequence (with higher min-entropy) we may approach the theoretic bound arbitrarily. However, this may (depending on the particular hash function used) increase the computational complexity. Also, if we use a part of our output as the d auxiliary bits or we reuse original randomness, the quality of the output distribution decreases, but by a limited and well-defined amount.

Theorem 4.7 ([SZ99]). *Let* $X_1, X_2, \ldots, X_l$ *be independent random variables defined on the sample space* $\mathcal{S} = \{0,1\}^n$ *with a common probability distribution*

p having min-entropy $H_\infty(p) \geq k$, $H = \{h|h : \{0,1\}^n \to \{0,1\}^{k-2e}\}$ be a universal$_2$ class of hash functions. Let $x_i \in_R \{0,1\}^n$ be randomly chosen from $\{0,1\}^n$ according to X_i and h be randomly and uniformly chosen from H. Then the distribution of $(h, h(x_1), \ldots, h(x_l))$ is $l\,2^{-e}$ close to the uniform distribution in the trace distance, i.e. l repeated applications of a fixed function randomly chosen from H is a $k, l\,2^{-e}$ strong randomness extractor.

4.3 Choosing Suitable Class of Hash Functions

A straightforward observation shows that any *XOR universal* class of functions [Sho96] is universal$_2$ as well, and, therefore, we may use any of the classes of hash functions designed for message authentication and heavily optimized towards computational efficiency. One should be careful when comparing computational efficiency, since in the case of message authentication only one (secretly) chosen hash function is being computed and therefore message authentication enables efficient precomputation we can rely upon only when the hash function is fixed and reused.

In our tests we used the universal$_2$ class of functions based on shift registers proposed in [IZ89]:

Let $r = r_1 r_2 \ldots r_n$, $x = x_1 x_2 \ldots x_n$, $x, r \in \{0,1\}^n$. Let $r \cdot x$ denote the scalar product of r and x, i.e., $r \cdot x = 1$ if the parity of $XOR(r, x)$ is odd and is equal to 0 otherwise. Let $r^{(l)}$ denotes the left bit rotation of r by l positions, i.e., $r^{(l)} = r_{l+1} r_{l+2} \ldots r_n r_1 r_2 \ldots r_l$. We define the m-convolution of r on x as the bit string $r(x)_m = r \cdot x, \ldots, r^{(1)} \cdot x, \ldots, r^{(m-1)} \cdot x$.

Let p be a prime such that 2 is a primitive root modulo p, let $n = p - 1$. Let $C_{n,m} = \{h_r | r \in \{0,1\}^p\}$, where $h_r : \{0,1\}^n \to \{0,1\}^m$ and $h_r(y) = r(1 \circ y)_m$ with $\circ$ denoting the concatenation of bit strings. The class of functions $C_{n,m}$ is universal$_2$ [IZ89].

Implementation of this class is efficient and straightforward – all necessary calculations are scalar product (i.e., bitwise XOR and mod 2 addition) and bit string rotation.

It is important to set suitable parameters for our application. As a final output of the extractor we would like to obtain a random sequence that is at least $\epsilon = 2^{-64}$ close to the uniform distribution in the trace distance, however, we would like to reuse the initial randomness (from the Nokia N73 digital camera) a number of times. For purposes of our testing we use less than 2^{20} repetitions (what suffices for more than 100 days of extracting), however, even doubling the exponent to 2^{40} implies only a loss of extra 40 bits per extraction.

Observing Theorem 4.7 we set parameters $e = 84$ and $p = 839$ (for efficiency reasons sufficiently large suitable prime number) what gives $m = k - 2e = \lfloor log_2 14 \cdot 838/4 - 168 \rfloor = 629$, since we expect that the min-entropy of our source is at least $\log_2 14$ and we have $838/4$ four-bit samples (see Section 5). Thus, from each 838 bits of input we extract 629 bits of output (compare to the theoretic optimal value 797 bits) giving the bit rate approximatively 36 bits of key per second.

5 Analysis of Acquired Random Data

To verify the input of the extractor and its min-entropy we ran a number of statistical tests on outputs of the function f. To obtain acceptable quality of the tests on an achievable number of samples (we used 1.5 mil. sample frames) we output only 4 bits from each frame (16 different values). We always tested the output for data captured at 8, 20 and 45 degrees Celsius.

First battery of tests concentrates mainly on the statistical *stability* of the output distribution – i.e., that the output data have all time (even in different external conditions) almost the same distribution (not necessarily uniform). We applied Pearson's (χ^2) goodness of fit test on several sequences and subsequences. As the expected distribution we take the relative frequency of 4-bit values from the whole sequence generated from all 1.5 mil. sample frames – in our case given by values: 0.0630, 0.0608, 0.0674, 0.0629, 0.0607, 0.0697, 0.0639, 0.0607, 0.0642, 0.0606, 0.0573, 0.0580, 0.0642, 0.0627, 0.0612, 0.0625. As the observed distribution we use the relative frequency of the 4-bit values from various subsequences – in our case, e.g., given by values: 0.0642, 0.0642, 0.0706, 0.0645, 0.0545, 0.0691, 0.0621, 0.0594, 0.0624, 0.0624, 0.0561, 0.0542, 0.0624, 0.0585, 0.0606, 0.0645. The resulting p-value (in this particular case 0.22) at the significance level $\alpha = 0.01$ clearly confirms our hypotheses that the sequences are at 99% from the same probability distribution.

Consulting the relative frequency we obtained (at the normal temperature of 20 degrees Celsius) we see that the min-entropy of tested distribution is approx. 3.984, what is almost the maximum of 4. As a further evidence of high entropy of our source we calculated the minimum of the Pearson's χ^2 test of our estimated distribution over all distributions with min-entropy at most $\log_2 14$. Recall that we designed the extractor the way it works for any distribution with min-entropy at least $\log_2 14$. We calculated

$$\min_{H_\infty(q) \leq \log 14} \chi^2(p, q)$$

with p being our estimated distribution. The outcome is $\chi^2 = 338.9947$, p-value is 2.0167×10^{-63}, what strongly rejects such a hypothesis.

As a part of our testing we considered other functions f as well. While it is easy to see that XORing an extra random bit to a given random bit can only decrease possible bias (or preserve it when the bit is deterministic), it would be nice to use as few input bits as possible. The reasons for such an optimization is not only the computational efficiency, but allocation of resources as well. As already mentioned, in our testing we used only a 4 bit output, but it might be useful to use just a few input bits to design extractors with higher bit rate, where independence of input data is argued in the terms of hardware. Our analysis showed that green color behaves in general more deterministically than red and blue color. Also, XORing a few pixels together already helps significantly to obtain higher entropy probability distribution. When comparing function f as described in Section 3, it has only a negligible (improving) impact if we XOR all bits from the pixel color channel instead of using only LSBs. Therefore, a

suitable function f should XOR together bits from several pixel distributed over the frame and use data from all color channels. On the other hand, it is sufficient to use LSBs only.

We also verified the design and our implementation of the randomness extractor (described above) by several statistical tests of the output. In this case we applied the well-known NIST statistical test suite to 40 MB output. All sequences passed 15 out of 16 statistical tests (frequency test, runs test, etc.). We tested division to 100 subsequences at the confidence level $\alpha = 0.01$. Passing the test means that a large fraction of subsequences passed particular sub-tests and also that the resulting p-values have uniform distribution. The Chi-square (χ^2) goodness of fit test is used to test the uniformity as well, but the confidence level is more strict in this case ($\alpha = 0.001$). The only one exception was "the serial test" that failed due to the small fraction of passing subsequences – their fraction 0.9500 was lower than (for 100 subsequences) the expected 0.960150.

6 Conclusion and Future Work

In our paper we describe a true random number generator delivering a bit string distributed according to a probability distribution very close to the uniform probability distribution. The initial randomness is acquired partly as an initial true random bit string and partly as a processed output of a built-in camera. In the previous analysis we used only a negligible portion of randomness contained in each frame acquired from the camera. This was only to allow for a reasonable statistical testing of data, while in practice much larger amount of data can be obtained from each frame. This would increase the bit rate significantly.

Another limitation is the sampling rate of 12 frames per second that can be increased provided independence between data produced from consecutive frames is guaranteed, e.g., through a hardware and postprocessing analysis.

Another area to explore is to select an optimal randomness extractor from computational complexity point of view, while preserving other required properties – near-optimal length of the extracted bit string and good behavior of the extractor when true random data are replaced by almost random data (or the same true random data are used many times). Future research will also concentrate on implementation of various randomness extractors within the mobile environment and comparison of their performance.

It is of an independent interest to investigate physical abilities of the adversary to influence the output of the CCD/CMOS chip at both precise and wide-area scale, as well as hardware and camera postprocessing based dependence between individual pixels within a single frame and between consecutive frames.

Acknowledgments. We acknowledge the support of the research project of Czech Science Foundation No. 102/06/0711 and the research project MSM0021622419. Jan Bouda also acknowledges the support of the project GAČR201/06/P338 of the Czech Science Foundation. Authors thank Luděk Smolík for stimulating discussions and Filip Jurnečka for his help with the extractor implementation.

References

[BST03] Barak, B., Shaltiel, R., Tromer, E.: True random number generators secure in a changing environment. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 166–180. Springer, Heidelberg (2003)

[CW79] Carter, J.L., Wegman, M.N.: Universal hash functions. Journal of Computer and System Sciences 18, 143–144 (1979)

[GW96] Goldberg, I., Wagner, D.: Randomness and the Netscape Browser. Dr. Dobb's Journal, Special issue on Encoding: Encryption, Compression, and Error Correction (1996)

[IL89] Impagliazzo, R., Levin, L.A., Luby, M.: Pseudorandom generation from one-way functions. In: Proceeding of the 21st ACM Symposium on Theory of Computing (1989)

[IZ89] Impagliazzo, R., Zuckerman, D.: How to recycle random bits. In: Proceedings of the 30th IEEE Symposium on Foundations of Computer Science, pp. 248–253 (1989)

[KMC07] Klingsheim, A.N., Moen, V., Hole, K.J.: Challenges in Securing Networked J2ME Applications. Computer 40, 24–30 (2007)

[KSM07] Krhovjak, J., Svenda, P., Matyas, V.: The Sources of Randomness in Mobile Devices. In: Proceeding of the 12th Nordic Workshop on Secure IT Systems, Reykjavik University, pp. 73–84 (2007)

[OS07] Lowe, G.: OpenSSL Security Advisory – OpenSSL FIPS Object Module Vulnerabilities (2007), http://www.openssl.org/news/secadv_20071129.txt

[DS08] Bello, L.: Debian Security Advisory – OpenSSL predictable random number generator (2008), http://www.debian.org/security/2008/dsa-1571

[Sha02] Shaltiel, R.: Recent Developments in Explicit Constructions of Extractors. Bulletin of the EATCS 77, 67–95 (2002)

[Sho96] Shoup, V.: On fast and provably secure message authentication based on universal hashing. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 313–328. Springer, Heidelberg (1996), http://www.shoup.net/papers/macs.pdf

[SMH05] Simonsen, K.I.F., Moen, V., Hole, K.J.: Attack on Sun's MIDP Reference Implementation of SSL. In: Proceeding 10th Nordic Workshop on Secure IT Systems, Tartu University, pp. 96–103 (2005)

[SZ99] Srinivasan, A., Zuckerman, D.: Computing with very weak random sources. SIAM Journal on Computing 28(4), 1433–1459 (1999)

Towards Modelling Information Security with Key-Challenge Petri Nets

Mikko Kiviharju[1], Teijo Venäläinen[2], and Suna Kinnunen[2]

[1] FDF Technical Research Centre, P.O. Box 10,
FIN-11311 Riihimäki, Finland
mikko.kiviharju@mil.fi
[2] University of Jyväskylä, P.O. Box 35 (Agora),
FIN-40014 University of Jyväskylä, Finland
teijo.venalainen@jyu.fi, suna.kinnunen@helsinki.fi

Abstract. Our global information society is based on distributed wide-area networks. Network security consists of the provisions made in an underlying computer network infrastructure, policies adopted by the network administrator to protect the network-accessible resources from unauthorized access, as well as continuous monitoring and measurement of the network security's effectiveness. In this paper, we describe the use of Petri nets in modelling network security. We propose a new hierarchical method for modelling network attacks and evaluating effectiveness of the corresponding defences. Our model is called Key-Challenge Petri Net (KCPN).

Keywords: Information security, Modelling, Petri nets, Attack Graphs.

1 Introduction

We are facing a tremendous growth in network traffic as business and service providers are turning to web based services. As the number of the provided services grows, the various information system architectures turn into service oriented architectures (SOA). This means, among other things, that there will be more and more network traffic not controlled by humans but automatic processes such as computer programs.

A great deal of research on information security (infosec) and network intrusions naturally accompanies this growth. There are also several different approaches on modelling infosec [30]. The most popular approach is to generate an attack graph in which attacker's actions, network states, and vulnerabilities are represented with nodes and arcs.

Attack graphs have several strengths; they are usually very straightforward to create and they can be extremely illustrative for a system administrator, depicting clearly the machines and the vulnerabilities the administrator needs to patch. A full attack graph contains all the possible attack sequences, so it provides all the needed information for threat analysis. However, due to the multitude of machines, and vulnerabilities on each machine, the attack graph becomes easily too complex to be analyzed or visualized, as in [31].

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 190–206, 2009.

On more abstract level, information security can be viewed as an access control matrix, which is an approach favoured in computer sciences [15]. However, this formalization can be *too* general and abstract, in which case straightforward implementations are either undecidable [5] or not expressive enough [6].

There are several other models between these two extremes of abstraction, but there seems to be a lack of a model which is both expressive enough to describe real world problems of realistic size and yet feasible to simulate or ask relevant questions. We tackle this problem here with *hierarchy*, which gives us the freedom to select our precision level within the model – theoretically, with enough simulation (=computing) power, infosec could be simulated down to individual hardware circuits, or very coarsely at subnet level.

As one definition of information security is a desirable set of states of an information system [5], together with a set of legitimate transitions between them, we can model infosec with state machines. However, the inherent concurrency and non-determinism of practical information security scenarios cannot directly be represented as sequential state machines. Mappings from such scenarios to basic state machines exist, but lead to a combinatorial explosion of the representation size.

An extension to state machines, called Petri nets (or P/T-nets – Place/Transition nets), allows a direct representation of concurrency and non-determinism within a set of states and is more readily translated into a model from a real-life scenario.

We are proposing an application and a special case of a generalization of stochastic Petri nets, called Stochastic Activity Networks (SAN), to simulate and study a wide variety of information security scenarios, encompassing the functionality of standard attack graphs as well as game-based approaches. Our approach combines the abstraction level of Key-Challenge Graphs (KCG, [7]) to hierarchical, coloured SANs (HCSAN, [2]). The approach is called Key-Challenge Petri Net (KCPN).

The contributions include:

- combining the principles of key-challenge graphs (KCG) [7] into an application of coloured and stochastic Petri nets,
- extending the key-challenge concepts to model:
  - all basic security attributes: confidentiality, integrity and availability
  - nodes (places) to incorporate more abstract entities than physical devices
- providing an intermediate abstraction level, yet powerful and easy-to-use model for simulation of technical computer security problems,
- incorporating game-based view to model both attacker (such as attack nets in [21]) and defender views (threat nets in [23]) to the model,
- ways to incorporate several approaches into one model through hierarchy.

In addition to the KCG, our concept incorporates ideas from the "*requires-provides*" model applied by Templeton et al. [33] for modelling computer attacks. We combine the ideas of above studies with the well studied structure of Petri nets. We use hierarchy as node aggregation and lifting the level of abstraction to control the size of graphs.

The remainder of this paper is organized as follows. Section 2 is a preview of related work in modelling information security and the KCG concept. In Section 3, we will look into the requirements we would like our model to fulfil. In sections 4 and 5

we introduce our model, KCPN, via its basic building block, HCSAN. Section 6 is reserved for conclusions and issues for further research.

2 Related Work

Information security modelling has been performed from a number of viewpoints. The most relevant for this work are state machine models (including other types of Petri nets), goal-directed programming languages, key-challenge graphs, and attack graphs.

Goal-directed programming languages, such as Golog in [10] are used to simulate intelligent behaviour of attackers. This has relevance in the decision making procedure of choosing different routes, but does not address the problem as widely as we need.

State machine models, especially those by Goseva-Popstojanova et al. in [11] and [19], are by far the closest to our general desired approach. They define a nine-state state machine to simulate any system's reactions to information security operations and attacks. This model is effective but rather simple.

An attack graph in infosec is typically a directed graph where nodes without outgoing edges depict the goals of the attack. The vertices represent a state of attack, i.e. effects of the attack so far (for example user access level on hosts, installed Trojans etc.), and the (directed) edges are transitions from one state to another.

It is possible to construct an attack graph that includes all the possible atomic attacks in the system. However, the resulting graph becomes easily too large to handle. The issue of state explosion has been addressed in different ways. Methods include, for example, node aggregation [25] (concerning identical network machines, vulnerabilities etc.), and concentrating on causality [1, 32, 33, 7, 35, 16, 27, 8].

A logical method to keep the amount of information manageable is to combine hosts with identical reachability. Ingols, Lippmann and Piwowarski presented in [13] a grouping of hosts into intra-subnet and inter-subnet reachability groups which led to 60% reduction of the complexity of reachability information in a small example network, according to the authors. The complexity of the information on network devices may be decreased in the same manner. Identical devices (e.g. same vendor, model, software, vulnerabilities etc.) may be grouped together.

Various studies share some common parameters for determining security. Some assign costs [32, 26] or probabilities [22, 34] to the attacker's actions (i.e. nodes or arcs).

Attack Graphs can be constructed using off-the-shelf model-checking tools [22] or other modelling methods, such as Petri nets (see e.g. [21, 12, 17]). Also various extensions of Petri nets may prove to be powerful instruments when modelling probabilities, time-depended activities, or multiple simultaneous attacks. Related work includes schemes such as Hidden Coloured Petri Nets (HCPN) [35] which, like our construction, also use coloured Petri nets for intrusion detection and integrate probabilities into the firing of the transitions.

Key-challenge graphs (KCG) were defined in [7] to originally model insider attacks, i.e. a phase after the reconnaissance, where the attacker already is aware of the structure of the network. It is also assumed that a successful compromise of a target is

not possible without a channel of interaction. There may be strong security mechanisms in place, which could force the attacker to consider alternative routes to reach the target. Each sub-target requires an additional effort but can provide the attacker information/capability and another "front", from which to continue the attack.

Figure 1 shows the basic building block of a key challenge graph. Any physical entity with information or capability that can be acquired is presented as a *vertex* of the graph. A vertex can be, for example, a database server or a computer system. Each piece of information or capability that is in a vertex is presented as *key*. A key could be, for example, a password stored on a computer. A channel of access or communication between two physical entities is presented as a *directed edge* between the two corresponding vertices. If there is a security measure or an enforced security policy protecting the resource it is denoted by a *key challenge*. A key challenge could be, for example, a password authentication required prior accessing to a server. If a user does not have the proper key to the challenge (key_w in Fig. 1) he incurs a significant *cost* (c_2 in Fig. 1) in breaking or circumventing the security measure. If a user has the proper key the cost is small or non-existent (c_1 in Fig. 1). Once the user has traversed from vertex *u* to *v* he receives the key_v. The starting point of an attack could be one or more vertices in the graph. These vertices are assumed to be initially in the control of the attacker. Attacker can also have multiple targets and his goal is to compromise them all.

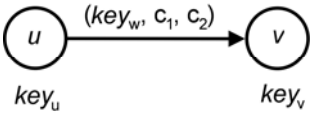


Fig. 1. Two key challenge graph nodes with a vertex

3 Requirements

In order to evaluate our model, we mapped out the requirements the model should fulfil in accordance to the questions posed for the conventional information security simulation models. The related work includes attack graphs, state machines, packet wars, network simulation tools, and canned attack/defend scenarios. These requirements were first presented in [14].

We adopted a "trinity" of abstract requirements, assuming that all practical security policies can be defined in terms of preserving confidentiality, integrity and availability (CIA) of information security entities, data, and metadata. Each time a CIA-requirement is not fulfilled, even temporarily, we say there has been a security breach. An ideal information security model will be able, in theory at least, to register all the (information systems) security breaches and report on the desired security statistics of the modelled system.

Our requirements are divided into three core requirements of registering (**REG**), statistics (**STAT**), and causality (**CAUS**) and two optional requirements of time-efficiency (**TIME**), and ability to represent dynamic situations (**GAME**). **REG**, **STAT** and **CAUS** are the more fundamental requirements of the model. The two

additional, requirements consider the efficiency and dynamic nature of the attack game. We note that a model meeting all the requirements might be impossible to achieve (due to the complexity of analysis), but, nevertheless, they serve both as a yardstick and a guideline in the construction of our model. The requirements are defined as follows:

REG: The model is required to register all the violations of confidentiality, integrity and availability (CIA) in reasonable time and within the boundaries stated by **STAT** requirements (see below). The model is not expected to recognize security breaches in every system, but if it does detect one breach it is required to detect all of them. The model should either: 1) report failure in polynomial time respect to the size of the definition of the modelled system, *or* 2) enable efficient representation of the system in relation to its complexity and size. Here the efficiency is regarded in relation to the available simulation or computing power. In practice this means the model should be scalable, which can be achieved e.g. through hierarchy.

STAT: The model should gather statistical information. This requirement is crucial to the security of the system as well as to the efficient concentration of the defender's resources. The system's representation is assumed to be given and the forbidden states and objects to be identified. The representation combines resources of both the attacker and the defender with different components of the model. Five important resources have been identified: time; computing power; available security tools and technical information security; controlled network (the model may incorporate network outside the immediate control of the defender's and attacker's); abstract workload of the defender to change the attributes of the modelled system.

Using the statistical information gathered, the model should give the probability of the system to be compromised in function of time and the probability of the system to stay in the compromised state in function of time.

CAUS: The model should have causality that enables recording the simulation path and re-simulation.

TIME: Using the model should be efficient for users equipped with varied computing power. The efficiency should include: formation of the description of the system to be modeled; analysis of the system to be modeled; employment of changes and remodeling; performing statistical functions (**STAT** requirement)

Especially the statistical functions should be efficient to perform so that preliminary or estimated results of higher level approximations would be available.

GAME: A typical information security breach or attack is interaction between the attacker and defender and is therefore better described as a stochastic game rather than a deterministic state machine. To fulfil the **GAME** requirement, the model should be able to include different kinds of countermeasures, even counter attacks. The model should allow inclusion of operation and properties that are not part of the modelled system but integral part of the attack. The skill-level of the attacker or the defender and manual power cycling serve as an example.

Further development of the **GAME** requirement could include self-modification of the model. The model could estimate functioning of an unknown element, such as an outside network or reconnaissance, and update the estimated parameters into the description of the system. We call this requirement **GAME***.

4 Stochastic Activity Networks

In the information security the most important aspect of the Petri nets is the ability to study reachability: whether a certain marking (initial state, which may or may not be secure) can lead to a set of other, undesirable markings (insecure state, less secure than the initial state).

The reachability problem in general is **EXPSPACE**-hard [18] (requiring exponential space, several complexity classes beyond **NP**-hard) although it is decidable [20]. Even though important special cases are efficient (polynomial) to solve, within information security, even in the relatively simple cases, the required resources tend to grow exponentially. This is one of the main reasons we investigated ways of reducing the Petri net size in order to improve efficiency together with functionality.

Due to the wide applicability of Petri nets, several extensions and decompositions to the basic Petri nets have been proposed. Our approach needs to distinguish between several types of tokens, as well as to measure different resources (most importantly time), and to act probabilistically. A generalization to Petri nets is a stochastic activity network (SAN). It also has hierarchical and "colourized" extensions we use in our construction.

SANs were first introduced in 1984 [24] to model concurrency, timeliness, fault-tolerance and degradable performance, all properties associated with symptoms occurring in information security attacks. More recently they were extended to a direction essential to our model, hierarchy [4]. In a paper by Movaghar and Azgomi [2], the "colourized" model was presented, called coloured SAN (CSAN). As CSAN is also a hierarchical model, we underline the hierarchy property by explicitly naming it hierarchical, coloured SAN, HCSAN.

According to [2] and [4], the HCSAN and (correctly formulated) hierarchy can be represented as a 16-tuple together with some additional definitions:

$HCSAN = \langle \Sigma, V, P, CP, IA, TA, MA, IG, OG, IR, OR, CF, C, F, \Pi, \rho \rangle$, where:

- Σ is a finite set of token types (colours).
- V is a finite set of global variables.
- P is a finite set of simple places (with non-coloured tokens)
- CP is a finite set of coloured places, where each $cp \in CP$ is a pair $\langle TT, SS \rangle$, where $TT \in \Sigma$ and $SS \in \{NONE, FIFO, LIFO, PRI\}$ is a token removal prioritization type, and P is disjoint with CP.
- IA is a finite set of instantaneous transitions/activities, which may fire as soon as the enabling conditions are set, non-deterministically if several are enabled at the same time; instantaneous activities take precedence over timed activities.
- TA is a finite set of timed transitions/activities, which at the time of enablement become active and remain so until completion (after the random time

interval specified by the marking- and place-dependent probability distribution function); only one timed activity is allowed to complete at a time and the order of completion for multiple enabled timed transitions is decided stochastically; a rate function specifies how fast the activity occurs.

- MA is a finite set of macro activities. Macro activities enable the hierarchy within the model and they are defined recursively as an 18-tuple $MA = \langle \Sigma, V, P, CP, IA, TA, MA, IG, OG, IR, OR, CF, C, F, \Pi, \rho, IFP, OFP \rangle$, where the first 16 items are as defined in HCSAN – practically a recursive definition of a HCSAN – and the additional components as:
  - IFP is the list of input fusion places, such that $IFP \subseteq P \cup CP$
  - OFP is the list of output fusion places, such that $OFP \subseteq P \cup CP$, and $|IFP| + |OFP| > 0$

  Fusion places integrate the sub-graph SAN with the higher level SAN such that there is a function $F_f: MA \times (IFP \cup OFP) \rightarrow P \cup CP$, which keeps the places in the MA at the same status as the corresponding places in the higher level of hierarchy. Theoretically, entering the "subroutine" can occur at the same place as exiting (that is, input- and output-fusion places may be the same node in the higher level hierarchy, as long as re-entering the MA is either allowed or ruled out with the enabling conditions.
- IG is a finite set of input gates with m (finite number of) inputs. $\forall(g \in IG)\exists(f_{IG}: M_1 \times \ldots \times M_m \rightarrow M_1 \times \ldots \times M_m \wedge g_{IG}: M_1 \times \ldots \times M_m \rightarrow \{\mathbf{true}, \mathbf{false}\})$, where f_{IG} and g_{IG} are the function and enabling predicate of the input gate, respectively, and $M_i = \mathbb{N}$ if $P_i \in P$ and $M_i = L(P_i)$, if $P_i \in CP$. Here $L(P_i)$ means the set of all possible instances of a list of items of type P_i. The function represents the change in the current marking caused by firing of the activity, and the enabling predicate defines when the activity is ready to fire.
- OG is a finite set of output gates with m (finite number of) outputs. Each output gate has a function defined like in IG, but without predicates.
- $IR \subseteq (P \cup CP) \times \{1, \ldots, |P \cup CP|\} \times IG \times (IA \cup TA \cup MA)$ is called the input relation for input gates, and they define the arcs from places to activities via the input gates. Formally, they have to fulfil the following conditions (assume $g \in IG$ has m inputs):
  a) $\forall(\langle P_1, i, g, a \rangle \in IR): i \leq m$
  b) $\forall\big(g \in IG \wedge (i \in \mathbb{N}, i \leq m)\big)\exists(a \in (IA \cup TA \cup MA) \wedge P_1 \in P \cup CP): \langle P_1, i, g, a \rangle \in IR$
  c) If for any two $\langle P_1, i, g_1, a \rangle, \langle P_2, j, g_2, a \rangle \in IR$, then it must be that $i \neq j$ and $g_1 = g_2$
- $OR \subseteq (IA \cup TA \cup MA) \times OR \times \{1, \ldots, |P \cup CP|\} \times (P \cup CP)$ is called the output relation for output gates, and they define the arcs from activities to places via the output gates. Formally, they have to fulfil the same uniqueness conditions as the input relations.
- $CF: CP \rightarrow \Sigma$ is a colour function that maps each coloured place to a type. This means that each token must have a data value of known type / colour.

- $C: M_1 \times ... \times M_m \times IA \rightarrow [0,1]$, where the activity is assumed to have m inputs and M_i defined as above, is the marking- and instantaneous activity-specific case probability function, which is used to define the probability, which of multiple concurrent enabled instantaneous activities will fire.
- $F = \{ F(.|\mu, a)\}$, where $\mu \in M_1 \times ... \times M_m$ is a marking and $a \in TA$ is a timed activity, is the set of marking- and (timed) activity-specific activity time (probability) distribution functions. The functions are used (as explained with TA) to decide probabilistically, how long timed activities will complete while active.
- $\Pi: M_1 \times ... \times M_m \times TA \rightarrow \{\texttt{true}, \texttt{false}\}$, where the M_i and m are defined as above, is the marking- and (timed) activity-specific reactivation predicate, which makes a timed activity active again after its completion (if it still is enabled).
- $\rho: M_1 \times ... \times M_m \times TA \rightarrow \mathbb{R}_+$ is the marking- and (timed) activity-specific enabling rate function, which defines how fast tokens are consumed from its input places, while the timed activity is completing (active).

Additionally, for HCSAN, we define the marking function, which will assign tokens to different places:

- $\mu: (P \cup CP) \rightarrow M$, is a marking function, assigning a number of tokens to each state / place. The marking of a whole HCSAN means a listing of the markings in all of the places. A marking is enabled, if for any of the places, the enabling predicate of its input gate evaluates to **`true`**. A marking is said to be stable, if no instantaneous activities are enabled, and unstable otherwise.

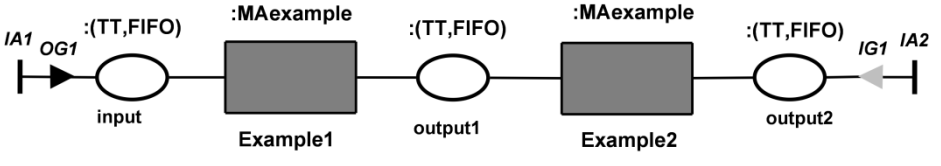


Fig. 2. Example-HCSAN with macro activities

To illustrate the properties of HCSAN, consider the example in figures 2 and 3. In fig. 2, **Example1** and **Example2** are macro activities of the HCSAN **:MAExample**. Places named **input, output1** and **output2** are coloured places which hold coloured tokens of **TokenType** (**TT**) type. These places are linked to input and output fusion places in macro activities. Tokens are removed from these places according a FIFO (First In First Out) scheduling policy. Hence the marking **:(TT,FIFO)**. The figure also contains normal input gates (**IG1**), output gates (**OG1**) and instant activities (**IA1** and **IA2**).

Macro activity class **MAExample** is illustrated in figure 3. It consists of input gates (**MA_IG1** and **MA_IG2**), output gates (**MA_OG1** and **MA_OG2**), coloured places (**place1**), timed and instant activities (**MA_TA1** and **MA_IA2**, respectively) as well as input and output fusion places (**input** and **output**, respectively). The input and output fusion places are tied to the upper level coloured places by the *MA* function F_f.

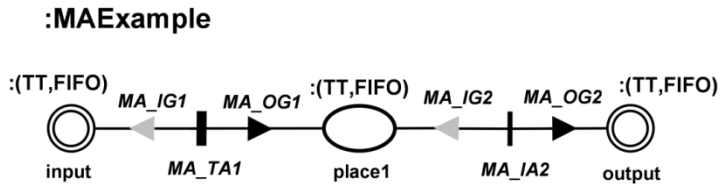


Fig. 3. A sample HCSAN macro activity

5 Key Challenge Petri Net

Our model is called the Key-Challenge Petri Net (KCPN). Formally, KCPN is an application for a modified HCSAN, where the crucial element includes interactions of attributes between tokens and places. Informally, KCPN models an active information security game as follows:

- Places represent various protected information security items; their supporting systems, from computer networks to individual cryptographic keys or even operating system state variables (depending on the level of abstraction), and information system states. Nets built from system state places are separated hierarchically from nets built from other types of places. Places may have one or several of the following properties: `Confidentiality`, `Integrity` and `Availability` (CIA). Places also have their "home colour" as a property / attribute.
- Tokens represent active players, or their influence, trying to access or deny access to certain places. Different colours of the tokens identify each party playing. Single-colour cases can be considered to represent a general coordinated attack either from the defender's or attacker's point-of-view (but not both). If a mismatch between colours of a place and a token happens, the CIA-properties defined for that place are considered lost or breached.
- Arcs between places and activities represent existing relations between protected items and their support systems.
- Input gates hold a list of mandatory keys for entering a challenge. In input gates it is decided whether the attacker has all the required keys for passing the associated challenge without any additional cost.
- Activity is the challenge attacker must answer before entering a place. In other words, activity models attack actions that allow him/her to gain needed information, access or resources. Challenge can be modelled as a function of time, computing power and other resources needed to complete the attack action.
- Output gates hold the keys the attacker receives after answering a challenge. Successfully passing the preceding activity, token moves into the output gate and is given the keys of the following output place.

An important concept in KCPN is the usage of *keys*. Formally, they are members of the power set of the places used as attributes within places, input gates and tokens. Informally, they represent access rights (e.g. a password) to pass a challenge, or

enable an input gate. Examples of keys include passwords, knowledge/exploit of vulnerabilities, or even Denial of Service (DoS) -attacks if it allows access to an `Availability`-node.

In a normal network state, due to the usual access restrictions, a token with a different colour from the home colour of the particular subset of the KCPN is allowed to traverse only through input gates signifying public access. However, if different-colour hostile tokens (attacker tokens) happen to have certain keys, they are allowed through. Keys are usually gathered from places, so that all the keys from a place accessed are considered to be available for a token visiting that place. However, new keys can also appear (if new vulnerabilities are discovered) and already acquired keys can be denied from the attacker (if vulnerability is patched).

5.1 Formalization

We formalize the KCPN based on HCSAN. We first need to redefine HCSAN to allow all of its main components to include structures. This applies to tokens, input gates and places, which need to have attributes (data items, such as keys). This extension does not require drastic changes to the formalization, as the token sets already can contain "complex" colours, i.e. structures of keys.

Mainly we need to update the perception on the definition of functions and adding attributes to places. Updating the places is not necessary at our phase of updated HCSAN yet: instead we add a separate key-assignment function in the main KCPN definition.

We use an intermediary concept, HCSAN^E, which differentiates our version from that in [2]. We omit the informal description of items. Formally, HCSAN^E is a 16-tuple exactly like HCSAN, except for the following items:

- CP does not contain prioritization types, as we have found no use for them outside communications simulation.
- Functions in HCSAN, including f_{IG}, g_{IG}, f_{OG}, g_{OG}, C, F, Π, ρ, and μ need to be understood as relations covering at least all logical expressions. It is possible that the resulting relations are not unambiguous, and thus not functions.

With the aid of HCSAN^E, we define the KCPN as an 11-tuple:

$$KCPN = (HCSAN^E, CIA_S, Mon_S, Mon_H, \mathcal{K}_{ey}, KF_S, KF_\Sigma, KF_{IG}, Res, Use_\Sigma, Use_{IG}),$$

where:

- $HCSAN^E$ is our slightly modified version of a hierarchical coloured SAN, defined above.
- $CIA_S: S \to 2^{\{\mathbf{C},\mathbf{I},\mathbf{A}\}}$, where $S = P \cup CP$, is the security function, assigning each place a set of information security attributes **C**, **I** and **A** (from confidentiality, integrity and availability) the place should be guarding, evaluates to an empty set, if the place performs a supportive function only.
- $Mon_S: S \to \{\mathbf{monitored}, \mathbf{not\ monitored}\}$, is the initial monitoring function, defining, which places are "interesting", i.e. whether and under what kind of resource consumption attackers can reach the monitored nodes.
- $Mon_H: S \cup MA \to \{\mathbf{breached}, \mathbf{secure}\}$, is the security monitoring function, defining, when a place or macro activity is considered secure or breached. For simple and coloured places this is a simple matter of translating security policy

to HCSAN place attributes, but for macro activities the function must be considered case by case as a combination of the individual places and macro subactivities within the macro activity at hand. This function is used to deliver **REG**-properties higher up the hierarchy.

- $\mathcal{K}ey \subseteq 2^S$ the set of possible keys; each key is a list of places that can be accessed with that key; note that in the actual simulation, keys can be given more descriptive names, binding them more closely to the application.
- $KF_S: S \rightarrow KS$, where $KS \subseteq 2^{\mathcal{K}ey}$, is the key assignment function for places that assigns a set of keys for each place.
- $KF_\Sigma: \Sigma \times Mk \rightarrow KS \subseteq 2^{\mathcal{K}ey}$, where Mk is the set of possible markings, is the key assignment function for tokens that assigns a set of keys for each token in a given marking. This function can be used e.g. to add and remove keys from the attacker as he advances in the model (provided the addition / removal models a reasonable action).
- $KF_{IG}: IG \rightarrow M_1 \times \ldots \times M_m \times KS$, where $KS \subseteq 2^{\mathcal{K}ey}$ and m is the number of input gates for the particular gate and the M_i are defined as $M_i = \mathbb{N}$ if $P_i \in P$ and $M_i = L(P_i)$, if $P_i \in CP$; and $L(P_i)$ means the set of all possible instances of a list of items of type P_i, is the key assignment function for input gates, specifying, how and which keys together with different types of tokens affect the firing controlled by the particular input gate.
- Res is the set of resource types used, each resource type is assumed to be measurable with a simple metric that can be described with a single real number.
- $Use_\Sigma: \Sigma \times Res \rightarrow \mathbb{R}_+$ is the resource use function for token types, which maps to each token type and resource type a real number signifying the current reserve of that particular resource type.
- $Use_{IG}: IG \times Res \rightarrow M_1 \times \ldots \times M_m \times \mathbb{R}_+$ is the resource use function for input gates, which maps to each input of an input gate and resource type a real number and an expression defining, how much traversing that input gate will use up resources of the particular resource type for tokens.

This formalization of KCPN captures the stochastic nature of the system, as well as the use of keys and the binding of different resources to the searching and using of the keys. The security function assigns places information of what kind of security functions they need to provide, if any.

The formalization hides many practical rules behind the different functions, particularly the key assignment and input gate functions and their expressions. Thus the actual expressive power of the model is best validated through an example application and an instantiation with an existing simulator. As it turns out, SANs are a well established model with simulators possible to use, for example METASAN [28], UltraSAN [29], SANBuilder [3] and Möbius [9].

As explained later, the KCPN extensions are mainly the infosec-application specific additional data for record keeping, which do not add substantial complexity to the CSAN model.

5.2 Example Behaviour

We now give an example of how KCPN can be utilized in modelling a network attack. Figure 4 illustrates a simple case in which there are two network devices, a workstation host and a database server that are interconnected. First, we have identified some attack methods which are typical for workstations and servers. Next, we have created macro activity classes for both workstation and server. In this example we use instances of both classes and they are described in separate graphs.

We have divided the whole system into two hierarchy levels: topology level (TL) and attack action level (AAL). TL represents the physical network connections between devices and AAL the actions by attacker in order to breach devices. In general, different level types can be interspersed freely: for example there could be several topology levels from network to individual device, then several attack action levels (like attack graphs) after which topology levels could continue detailing software modules and even hardware buses.

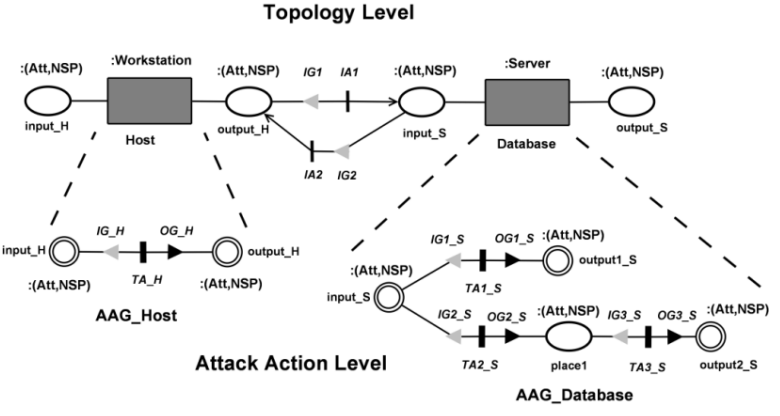


Fig. 4. A two-level Key-Challenge Petri Net

AAG_Host is a macro activity and an instance of the attack action graph class **:Workstation**. The class represents typical attacker's actions for every workstation. **AAG_Database** is an instance of class **:Server**. It should be noted that multiple instances of workstation and server macro activities may be used in constructing a KCPN model. However, the associated keys may differ between instances and therefore macro activities are not identical even though they look alike.

The keys for passing the challenge are checked at the input gates using the KCPN function KF_{IG}. On the topology level the keys represent permissions to establish a connection and in attack action level they represent information or access rights to circumvent a security measure. Activities represent attack actions and establishing connections in AAL and the TL, respectively. Connections can be made from either host into the database or vice versa, hence the two activities in the TL. Note that the TL activities are instant and AAL activities are timed. This is because our model tries to encompass general reconnaissance-phase and slower attacks, which may take substantially longer than establishing a connection between networking devices. Resource use for tokens is determined in input gates using function Use_{IG}.

Output gates give keys (using function KF_{Σ}) from output places to tokens. Places represent successful connections in the topology level and successful attack actions in the attack action level. The places may contain attacker-type tokens and there is no priority in the scheduling strategy for removing tokens from places, hence the marking **:(Att,NSP)** (see [2]). Other KCPN functions are used in our example as follows:

- CIA_S: Assigns a value for places (in this example only in AAL) that determines which security property (confidentiality, integrity or availability) is considered lost when a token reaches the place.
- Mon_S: Determines those places in either level that are monitored by the modeller. These are usually the ultimate goals of the attacker.
- KF_S: Assigns a set of keys for each place in the AAL.
- KF_{IG1_S}: Assigns the combination of keys to the input gate **IG1_S** that the attacker will need in order to traverse the activity **TA1_S** and to breach the security attributes in the node **output1_S**, which is the target database, and the end node of the AAL macro activity.
- Res: Defines the resource types used in the model.
- Use_{Σ}: Keeps track of each token's resource use.

In our example, the attacker token moves first into the **AAG_Host** through the **input_H** fusion place and back to the topology level through the **output_H** fusion place. Then the token moves through the **IA1** into the **AAG_database**. There the token may take two possible routes and breach the database.

5.3 Analysis

The KCPN is essentially an application of HCSAN. The most important points for analysis are:

- the addition of security keys to places
- a more complex function (in practice) for the input gates to evaluate
- unknown (possibly non-exponential) probability distributions for the timed activities

Of these, the first two points do not present theoretical obstacles for standard CSAN analysis techniques, state-space analysis with the reachability graph formulation into a continuous-time Markov chain [2] and simulation. However, the SAN technique on reducing the difficulty of the general reachability graph problem for Petri nets requires exponential probability distributions for the timed activities, and this is something we cannot currently guarantee.

We have conducted preliminary simulations of the KCPN model and found that the complexity is roughly $O(n^2)$, where n is the number of places. In the tests we used up to five hierarchy levels and the security monitoring function Mon_H, which communicates security-breach information across the modelling levels. However, it should be noted that we keep a strong emphasis on limiting the number of the places in the future work.

Essentially, KCPN adheres to the same analysis techniques as CSAN, and analysis can be made efficient by either:

- constraining the probability distributions of the TAs to be exponential,
- designing the hierarchies suitably for the available simulation power and most interesting nodes (by defining the initial monitoring function Mon_H properly).

We should note that there are several ways of taking advantage of the hierarchy. In addition to reducing the modelling complexity, the hierarchy introduces a way of simulating several approaches at once. Our example includes integrating several hierarchy levels based on topology and attack graph levels. These can be layered freely on top of each other, enabling harnessing of the efficiency of the attack graphs and the expressiveness of the Petri nets.

Attack graphs are easily modified into a Petri net, by replacing the arcs with simple Petri net arcs and transitions (with corresponding input gates and their enabling predicates), and modelling the tree traversal by having tokens move in the nodes.

It is instructive to see how and to what degree the KCPN fulfils the requirements specified above. For **REG** the model defines clearly a set of conditions that must be fulfilled in order to be able to say that the system has reached a state of a security breach. These states can be found through a state-space search of the underlying Petri net. If the states and conditions contain sufficient information, all of the breached states will be found. Additionally, the hierarchy makes the search feasible.

For **STAT**, the model defines a set of "monitored" places, and the SAN-base guarantees the proper use of time and probability, so they can be measured together with all of the resources mentioned in **STAT**.

In **CAUS**, if the system keeps a log of its actions, the temporal order of firing sequences should be easy to establish, and thus infer causal relations between different events.

For **TIME**, due to the hierarchy, KCPNs can be quick to analyse. However, this does not translate to efficiency given the problem size, as most of the tasks require superpolynomial running times. Composition and change management of the model are easy, as the model is hierarchical and intended to be mapped directly from a real-life situation. The resource consumption related to **STAT** is straightforward to measure, as long as the system keeps a log of its actions, as required in **CAUS**.

In **GAME**: The colouring of the tokens and the use of tokens altogether, allows the simulation of dynamic situations. It is not known, how resources or factors outside the model could be translated into e.g. keys. The model does not currently allow adding places or gates or modifying their relations during the simulation, so it does not fulfil **GAME***.

We thus conclude that the KCPN fulfils, as a model, all of the requirement categories, but there are gaps to be filled within the categories and some questions may remain infeasible to solve, as of yet.

6 Conclusion and Issues for Further Research

We have proposed an extension to coloured and stochastic Petri nets combined with key-challenge. We call our model the Key-Challenge Petri Nets (KCPN). KCPN extends the key-challenge principle to model all the basic security attributes and its variable abstraction level makes it suitable for simulation of diversity of computer

security problems ranging from technical problems to social engineering. The KCPN makes possible modelling from both the attacker's and the defender's point of view. In future work the model will be extended to include a game-based modelling of the attacker's and defender's actions with multiple independent or collaborating attackers. Research on modelling unknown (zero-day) vulnerabilities into the KCPN model is also needed.

Acknowledgements. We would like to thank Anneli Heimbürger and Kari Kärkkäinen from University of Jyväskylä for their valuable comments and insights in this work.

References

1. Ammann, P., Wijesekera, D., Kaushik, S.: Scalable, Graph-Based Network Vulnerability Analysis. In: 9th ACM Conference on Computer and Communications Security, pp. 217–224 (2002)
2. Azgomi, M.A., Movaghar, A.: Coloured Stochastic Activity Networks: Definitions and Behaviour. In: 20th Annual UK Performance Engineering Workshop (UKPEW 2004), Bradford, UK, July 7-8, pp. 297-308 (2004)
3. Azgomi, M.A., Movaghar, A.: A Modelling Tool for Hierarchical Stochastic Activity Networks. In: 11th International Conference on Analytical and Stochastic Modelling Technologies and Applications, ASMTA 2004 (2004)
4. Azgomi, A., Movaghar, A.: Hierarchical Stochastic Activity Networks: Formal Definitions and Behaviour. International Journal of Simulation, Systems, Science and Technology 6(1-2), 56-66 (2005)
5. Bishop, M.: Computer Security: Art and Science. Addison-Wesley Longman Publishing Co. Inc., Boston (2002)
6. Bishop, M., Snyder, L.: The Transfer of Information and Authority in a Protection System. In: ACM Symposium on Operating System Principles, pp. 45–54. ACM, New York (1979)
7. Chinchani, R., Iyer, A., Ngo, H., Upadhyaya, S.: Towards a Theory of Insider Threat Assessment. In: International Conference on Dependable Systems and Networks, pp. 108-117 (2005)
8. Dawkins, J., Hale, J.: A Systematic Approach to Multi-Stage Network Attack Analysis. In: Second IEEE International Information Assurance Workshop, pp. 48-56 (2004)
9. Deavours, D.D., et al.: The Möbius Framework and Its Implementation. IEEE Transactions on Software Engineering 28(10), 956-969 (2002)
10. Goldman, R.P.: A Stochastic Model for Intrusions. RAID (10), 199–218 (2002)
11. Goseva-Popstojanova, K., Wang, F., Wang, R., Gong, F., Vaidyanathan, K., Trivedi, K., Muthusamy, B.: Characterizing Intrusion Tolerant Systems Using a State Transition Model. In: DARPA Information Survivability Conference and Exposition (DISCEX II), pp. 211–221 (2001)
12. Helmer, G., Wong, J., Slagell, M., Honavar, V., Miller, L., Wang, Y., Wang, X., Stakhanova, N.: Software Fault Tree and Coloured Petri Net-Based Specification, Design and Implementation of Agent-Based Intrusion Detection Systems. International Journal of Information and Computer Security, 109–142 (2007)

13. Ingols, K., Lippmann, R., Piwowarski, K.: Practical Attack Graph Generation for Network Defense. In: 22nd Annual Computer Security Applications Conference, ACSAC 2006, pp. 121–130 (2006)
14. Kiviharju, M., Kinnunen, S., Kärkkäinen, K., Venäläinen, T.: State Machines for Information Systems Security. Lanchester and Beyond – A Workshop on Operational Analysis Methodology, FDF Technical Research Centre Publications no. 11, pp. 71–88 (2006)
15. Lampson, B.: Protection. In: ACM Operating Systems, Rev., vol. 8, pp. 18–24. ACM, New York (1974)
16. Li, W., Vaughn, R.B.: Cluster Security Research Involving the Modeling of Network Exploitations Using Exploitation Graphs. In: IEEE International Symposium on Cluster Computing and the Grid (2006)
17. Lin, C.-C., Wang, M.-S.: Depth Evaluation Intrusion Detection using Coloured Stochastic Petri Nets. In: Proc. of IEEE International Conference on Intelligence and Security Informatics, pp. 248–250 (2008)
18. Lipton, R.: The Reachability Problem Requires Exponential Space. Technical Report 62, Yale University (1976)
19. Madan, B., Goseva-Popstojanova, K., Vaidyanathan, K., Trivedi, K.: Modeling and Quantification of Security Attributes of Software Systems. In: International Conference on Dependable Systems and Networks, DSN 2002 (2002)
20. Mayr, E., Meyer, A.: The Complexity of the Finite Containment Problem for Petri Nets. Journal of the Association for Computing Machinery 28(3), 561–576 (1981)
21. McDermott, J.: Attack Net Penetration Testing. In: 2000 Workshop on New Security Paradigms, pp. 15–21 (2001)
22. Mehta, V., Bartzis, C., Zhu, H., Clarke, E., Wing, J.: Ranking Attack Graphs. In: Zamboni, D., Krügel, C. (eds.) RAID 2006. LNCS, vol. 4219, pp. 127–144. Springer, Heidelberg (2006)
23. Mirembe, D., Muyeba, M.: Threat Modelling Revisited: Improving Expressiveness of Attack Nets. In: European Modelling Symposium, pp. 93–98 (2008)
24. Movaghar, A., Meyer, J.F.: Performability Modeling with Stochastic Activity Networks. In: 1984 Real-Time Systems Symposium, Austin, TX, pp. 215–224 (1984)
25. Noel, S., Jajodia, S.: Managing Attack Graph Complexity Through Visual Hierarchical Aggregation. In: 2004 ACM Workshop on Visualization and Data Mining for Computer Security, pp. 109–118 (2004)
26. Ortalo, R., Deswarte, Y., Kaaniche, M.: Experimenting with Quantitative Evaluation Tools for Monitoring Operational Security. IEEE Transactions on Software Engineering, 633–650 (1999)
27. Ou, X., Boyer, W.F., McQueen, M.A.: A Scalable Approach to Attack Graph Generation. In: 13th ACM Conference on Computer and Communications Security, pp. 336–345 (2006)
28. Sanders, W.H., Meyer, J.F.: METASAN: a Performability Evaluation Tool Based on Stochastic Activity Networks. In: ACM/IEEE-CS Fall Joint Comp. Conference, pp. 807–816 (1986)
29. Sanders, W.H., et al.: The UltraSAN Modeling Environment. Performance Evaluation 24, 1–33 (1995)
30. Saunders, J.: Simulation Approaches in Information Security Education. In: 6th National Colloquium for Information System Security Education (2002)
31. Sheyner, O., Haines, J., Jha, S., Lippmann, R., Wing, J.M.: Automated Generation and Analysis of Attack Graphs. In: IEEE Symposium on Security and Privacy, pp. 273–284 (2002)

32. Swiler, L.P., Phillips, C., Ellis, D., Chakerian, S.: Computer-Attack Graph Generation Tool. In: DARPA Information Survivability Conference & Exposition II, DISCEX 2001, pp. 307–321 (2001)
33. Templeton, S.J., Levitt, K.: A Requires/Provides Model for Computer Attacks. In: 2000 Workshop on New Security Paradigms, pp. 31–38 (2000)
34. Wang, L., Islam, T., Long, T., Singhal, A., Jajodia, S.: An Attack Graph-Based Probabilistic Security Metric. In: Data and Applications Security XXII, pp. 283–296 (2008)
35. Yu, D., Frincke, D.: Improving the Quality of Alerts and Predicting Intruder's Next Goal with a Hidden Colored Petri-Net. Computer Networks 51(3), 632–654 (2007)

Security and Trust for the Norwegian E-Voting Pilot Project *E-valg 2011*⋆

Arne Ansper[1], Sven Heiberg[1], Helger Lipmaa[1], Tom André Øverland[2], and Filip van Laenen[3]

[1] Cybernetica AS, Estonia
[2] Acando AS, Norway
[3] Computas AS, Norway

Abstract. Early 2009, the Norwegian Ministry of Local Government and Regional Development (KRD) decided to start a procurement procedure for *E-valg 2011*, an e-voting pilot project for the municipal and regional elections of 2011. The Norwegian companies Computas and Acando formed a consortium together with the Estonian company Cybernetica, the company behind the successfull electronic elections in Estonia in 2005, 2007 and 2009. The consortium proposed a solution based on an open source approach on the one side, and the knowledge and experience in the field of Cybernetica on the other side. This paper discusses the security and trust aspects of the proposal put forward to the KRD.

Keywords: Electronic Voting Schemes, E-voting, Open source, E-valg 2011, Norway.

1 Introduction

Early 2009, the Norwegian Ministry of Local Government and Regional Development (KRD) decided to start a procurement procedure for *E-valg 2011*, an e-voting pilot project for the municipal and regional elections of 2011 and some local referendums in the same year. If this pilot project is a success, the project will be continued and extended to be used in the parliamentary elections of 2013, and all futures elections from then on.

KRD, and by extension the Norwegian government, has many reasons to try out electronic voting over the Internet. One obvious reason is the possibility to speed up the counting process of the elections. If all voters would vote electronically, the counting process could potentially finish within a few minutes after the closing of the polling stations. Another reason is to attract some specific segments of the electorate, in particular young voters, so that their participation in the elections could rise. Electronic voting also has the potential to let disabled people, like the blind, vote without the assistance of other people, e.g.

⋆ This research has been supported by the Estonian Science Foundation, grant #8058, and by the European Regional Development Fund through the Estonian Center of Excellence in Computer Science, EXCS.

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 207–222, 2009.

through the use of screenreaders. A reason that should not be neglected either is the possibility to run elections in a more cost effective way. Indeed, elections on paper may require much more time and money than electronic elections, and the vaste amount of paper that could be saved is enormous.

This paper gives an overview over the security aspects of the Internet voting application client. The client is only a small part of the complete election system, but nevertheless an important one since it will be the only part the general public will use directly and therefore have a strong opinion about. In addition, this paper discusses some thoughts on the trust of the general public in the electronic election system, as this will be of key importance to make the *E-valg 2011* project a success.

2 Security in the Voting Application

This section presents and discusses a number of voting schemes relevant for the Norwegian situation. They range from a completely open voting application that has little or no security against Trojans on a voter's computer, to a fully-fletched "blind" voting scheme, putting no trust at all in the voter's computer. In every section, we first describe the scheme and the user interface that implements the scheme. Then we analyze the security of the scheme, concentrating on how well it deals with Trojans. The main concerns we will deal with are confidentiality (guessing a voter's choice) and integrity (changing a voter's choice to a random party and changing a voter's choice to a particular party). After the security analysis we take a quick look at the usability aspects of the scheme and whether or not it can be implemented as an open source project.

2.1 Norwegian Situation

Before we dive into the different voting schemes, a few notes should be made on the specific Norwegian situation. These notes may explain why some assumptions in the discussion of the security of the different voting schemes are true in the Norwegian case, even though they wouldn't be true in the contexts of other countries.

- The principle of the double envelope won't be difficult to accept for Norwegians in general, since they are already used to double envelopes in the advanced voting period. Indeed, if a Norwegian voter decides to vote in advance, he puts his ballot inside a blank envelope, which in its turn is put inside another envelope with his name on it.
- Since it is assumed that all voters in Norway will have the opportunity to go to a polling station, either in advance or on the polling day itself, the problem of coercion and vote buying is assumed to be dealt with outside the electronic voting system.
- It follows from the previous point that confidentiality on the voter's computer isn't such a big issue. Indeed, if a voter wants to sell his vote, he will always be able to do so if the government decides to give him the opportunity to vote

over the Internet from the comfort of his own living room. The same can be said for voter coercion. The residual problem is the one of Trojans installed on the voter's computer, but it can be argued that the electronic hygiene of each voter at home isn't the responsibility for the Norwegian government (but sensibilization is). There are also many easier ways to find out which party a voter is likely to vote for, that work even in the absence of electronic voting over the Internet.
- By the time the *E-valg 2011* will be rolled out, it is assumed that a national PKI infrastructure, dubbed *eID*, will be in place. This infrastructure will provide all Norwegians who want to with an electronic ID card, together with a card reader so they can identify themselves on the Internet using the card. There are already other solutions operational, the two main competitors being *BuyPass*[1] and *BankID*[2].

2.2 Open Voting Scheme Using Double Envelopes

Description. This straight-forward approach to e-voting is taken from the Estonian e-voting system. The voter's computer is fully trusted in this system, the list of parties and candidates is the same for all voters in one district and the ballot to be cast is kept in cleartext in the computer's memory for a short time.

The following steps describe the voting process in the case of municipal elections:

- The voting application authenticates the voter to the voting server with his national ID (eID).
- The voting application receives a list of parties and candidates from the server via a secure channel; this list is not encrypted and the same for all voters in the same district.
- The voting application displays the list of parrties in a point-and-click interface, ordered randomly.
- The voter makes his decision by clicking on the party name, and clicks on the Next button to continue. He has also the option to vote blank.
- In the next step, the client shows the candidate list for the party he selected, and allows him to give a personal vote to as many candidates as he wishes. Additionally, he may also add write-ins.
- In the last step, a summary of the voter's vote is presented. If the voter is happy with his choice, he can click on the Next button to encrypt and digitally sign the ballot, and send it to the server.

General Security Issues. This scheme fully trusts the voter's computer and at some point in time the cleartext ballot is stored in the computer's memory.

Eavesdropping Trojans. It is fairly easy for a Trojan to detect the contents of the ballot from the memory. As the candidate and party names are known

[1] See http://www.buypass.no/ (accessed August 2009) for more information.
[2] See http://www.bankid.no/ (accessed August 2009) for more information.

beforehand and are the same for all voters in the same district, the Trojan can be 100 % sure for whom the particular voter is voting.

Trojans Making Random Changes. A Trojan that wants to perform a simple DoS attack can easily modify the plaintext ballot's contents.

Trojans Voting for a Particular Party. The Trojan can easily modify the plaintext ballot's contents. As the party and candidate numbers are well known the Trojan can put its own preferences in the ballot.

Other Security Issues. If an attacker has enough time to analyze the implementation of the voting application, and then to design and distribute the Trojan, then the described attacks can be scaled to very large numbers.

This particular scheme is also subject to attacks that try to compromise eID. If e.g. someone gets his hands on someone else's PIN codes, then he can write a Trojan that waits for the ID card to be inserted into the machine and then votes for the ID card owner. These types of attacks can not be executed on large scale though.

It would be possible to somewhat increase the security of this solution by using code obfuscation techniques in order to hide the ballot in memory. However, according to the current project plans, the solution will be on-line during the whole advance voting period. Since this advance voting period stretches over three months, the effect of any code obfuscation techniques will be negligible.

Usability. A friendly point-and-click user interface can be designed for this solution, showing both the list of parties and the list of candidates to the user.

Open Source. Making this solution open source renders it even more accessible to Trojans, as attackers can search the source code for weak design patterns and vulnerabilities.

2.3 Blind Voting Scheme

Description. This voting scheme, which may slightly remind of Okamoto's receipt-free voting scheme [1], eliminates the need to trust the voter's computer. In fact, the voter's computer never sees a single party or candidate list, as the voting process on the client computer is performed using anonymous codes, personalized for each voter, hence the name "blind" voting scheme. The personalized codes the voter uses to vote are printed on his polling card, and they are mapped to the common list of parties and candidates using a homomorphic encryption scheme.

The common list may look like this:

Red Party	101
Green Party	102
Blue Party	103

Voters Alice and Bob receive personalized polling cards showing the following codes:

Party	Alice's codes	Bob's codes
Red Party	234	135
Green Party	567	963
Blue Party	890	468

Internally, the server maps the codes for the voters to the parties in the following manner:

- 234 to the Red Party for voter Alice,
- 567 to the Blue Party for voter Alice,
- ...
- 135 to the Red Party for voter Bob,
- ...

Since the server uses homomorphic encryption, it knows how to map a code to a party, but doesn't know which party a particular voter voted for.

Before the election can take place, polling cards with the party codes have to be distributed to the voters over a secure second channel. This may be the postal service in the Norwegian case, but in other countries this may not be an option. Note that paper polling cards already are being distributed in Norway before every election, so this requirement doesn't represent an additional cost.

When the voter uses the Internet voting application client, she has to go through the following steps:

- The Internet voting application client authenticates the voter to the voting server using national ID (eID).
- The Internet voting application client displays an input box where the voter can enter the code for a party. The voter has to consult her polling card for the correct party code. Error detection is added to the party codes so that simple mistypings do not prevent the voter from casting her vote.
- When the party code is accepted, the client displays a list of anonymous numbers representing the candidates. The voter can select the numbers of the candidates of her preference, but she has to know in advance which numbers correspond to which candidates. If the client would present the user with a list of names, a Trojan could easily deduce which party the voter is voting for.
- Next the voter can enter write-ins. One way to do this is to enter the party code for the candidate's party from the polling card, and then select the correct candidate number for the candidate.
- When the write-ins are added to, the client shows a short summary, which again can only consist of party codes and anonymous candidate numbers.
- When the voter clicks on the Next button, the cleartext ballot is encrypted and digitally signed, and sent to the voting server for conversion using homomorphic encryption.

General Security Issues. This scheme requires no trust in the voter's computer at all. As all the input is personalized there is no way for a Trojan to know what the numbers entered by the voter really mean. However, for this scheme to work, a trustworthy second channel to distribute the personalized codes has to be in place.

Eavesdropping Trojans. Trojans can still eavesdrop on a voter's computer, but as long as there is no access to the voter's polling card, no information is gained by eavesdropping. There is a possible leak of confidentiality though: If the voter wants to be sure which order her preferred candidate has on a party list, and she uses a web browser on the same computer she is voting from to determine the ordering, then the Trojan may find out about the voter's political preference that way.

Trojans Making Random Changes. Trojans can still make random changes to the ballot, but they cannot be sure whether the change will be accpeted as a valid code for a party or not. Most probably the change will result in a spoiled ballot, leaving the DoS attack to e-voting an unsolved issue. Sending out receipts over a third channel may give feedback to the voter, but if the voter decides to revote as a consequence of the information in the receipt, the Trojan may gain some information about his guesses and have higher chances to succeed in spoiling the ballot the next time the voter votes.

Trojans Voting for a Particular Party. Trojans wanting to vote for a particular party have no way to proceed as each user has a specific polling card.

Other Security Issues. The scheme needs a second communication channel that has to be trusted as well. It is possible for the attacker to direct his efforts to this channel. If e.g. the postal service is used as a second channel, then malicious postal workers could:

- steal polling cards, resulting in a denial of service, since without polling card one cannot e-vote,
- forge polling cards with access to the originals: if the forging is done to change the codes of Party A to those of Party B, then all the voters for Party A would actually vote for Party B,
- forge polling cards with no access to the originals, resulting in a denial of service without the voters even knowing it,
- copy polling cards: in this case the results collected by an eavesdropping Trojan could later be analyzed by the attacker.

These attacks do not scale well, but on the level of a small community they might have important effects on the results. It's also possible to attack a particular voter.

Another issue that should be addressed is what will happen if a voter loses his polling card. Should the authorities issue a new polling card with the same codes, or should new codes be generated too?

If the second channel is another computer or mobile based, then other kinds of attack are possible.

It might also be problem that the ordering of candidates is in plaintext. A Trojan can always cancel out the fifth candidate on the list to avoid a certain candidate to be elected, or give an additional vote to every sixth candidate. Additional layers of coding for candidate order could be introduced to increase the security, but would also complicate the user interface utterly.

Usability. The user interface looks awkward to people who are used to simple and intuitive user interfaces. Additionaly, the user interface is of no use when a voter has lost his polling card. As we already mentioned, if an additional layer of coding for the candidate orders is introduced, then the interface gets even harder to use for an average voter.

Open Source. The solution can be fully open source since it wouldn't give an attacker any advantages.

2.4 Blind Voting Scheme Using Symbols

Description. This scheme is an attempt to increase the usability of the basic blind voting scheme described in the previous section. Instead of numbers, (neutral) symbols or iconic pictures are used and printed on the polling card. For voter Alice, this could e.g. look like this:

Red Party ⋆
Green Party ⊕
Blue Party ◇

Just like for the codes, these pictures differ from voter to voter. On the server-side there is 3-way mapping: picture-code-party/candidate number.

Again, polling cards with symbols for each party have to be distributred to the voters over a secure second channel, just like for the scenario with party codes. The voter now has to go through the following steps to submit his vote over the Internet:

- The Internet voting application client authenticates voter to voting server using national ID (eID).
- The Internet voting application client displays a list of symbols. The voter consults his polling card for the symbol matching the party of his preference, and points and clicks with his mouse on the symbol that resembles his choice.
- The Internet voting application client then displays a list of anonymous numbers that represent the candidates. The same considerations apply as in the previous scheme: the voter has to know in advance the numbers of the candidates he wants to vote for.
- When the user is done, the cleartext ballot is encrypted and digitally signed, and sent to the voting server who converts it via homomorphic encryption.

General Security Issues. The mapping of symbols to codes must be known to the Internet voting application client. It must therefore be sent to the voter's computer by the voting server, and a Trojan will be able to read it. Notice that the server should always send the same set of codes to the client, both valid and invalid. Otherwise a Trojan could send three or four requests to the server, and eliminate the codes that don't reappear in the responses as invalid. After a few times, the Trojan will be able to guess which codes are the valid codes for a particular voter.

Eavesdropping Trojans. Eavesdropping is not easier than in the basic blind voting scheme, since a Trojan can't gain any information about a voter's political preferences by observing the symbols he selects in the user interface.

Trojans Making Random Changes. As the Trojan sees all the codes associated to the voter, it can make a better guess than in the basic blind voting scheme. If 100 icons are used to present to the voter, and 10 parties participate in the contest, then he has a $10/100 = 10\,\%$ chance of picking a valid code at random.

Trojans Voting for a Particular Party. Voting for a particular party is easier than in basic blind voting scheme in the sense that the space from which the Trojan has to make a selection is drastically reduced. However, the Trojan has no information about which symbol belongs to which party for a certain voter, and therefore only has a $1/100 = 1\,\%$ chance of picking the right code for a particular party.

Other Security Issues. The same considerations about trust in the second channel apply for this scheme as for the previous one.

Usability. This scheme is slightly more usable than the basic blind voting scheme, because party icons can be recognized by illiterate persons as well. On the other hand, it has to be noted that for illiterate people it may be a difficult task to use the Internet voting application client anyway, e.g. because they have to authenticate themselves using the national ID. For visually handicapped people, however, this scheme will be less usable.

Open Source. Open sourcing a project using this scheme may have the disadvantage that the mapping between symbols and codes can be found out more easily than otherwise.

2.5 Blind Voting Scheme Using CAPTCHAs

There are several other ways to improve the basic blind voting scheme. One such approach would be to eliminate the second channel. Each voter still receives a personalized list of parties, but now in the Internet voting application client in a similar way as in the open voting application scheme. This allows for a more

user friendly interface and offers some sort of protection against eavesdropping and ballot modification. This protection is not very strong though:

- The names of parties and candidates and corresponding codes are in the computer's memory at the same time. Although the numbers are personalized, it is still possible to detect the mapping via memory analysis. It should also be noted that the presentation of the party names and codes is somewhat more complicated than in straightforward case.
- A Trojan on a voter's computer can see all party and candidate numbers available for the voter. A simple Trojan may in fact use it for a more advanced DoS: the voter's choice is altered and this time the new choice is accepted by the server because the Trojan picks the correct number from the computer's memory.

It is possible to overcome these weaknesses by using CAPTCHA technology. CAPTCHAs are mainly used to distinguish between humans and computers on websites. Some sort of challenge that is hard for computer to solve, but easy for humans, is presented to the user. Most applications of CAPTCHA use object recognition: the human brain is very good at telling visual objects such as typefaced letter apart from its background, but for a computer this task is time consuming, and no general working solution is known. If the answer to the CAPTCHA appears relatively fast, then we know we are dealing with a human, otherwise it might be computer. It should be noted that CAPTCHA technology, or more correctly CAPTCHA breaking technology, is developing relatively fast. Therefore, a scheme that is secure today might not be secure tommorrow.

Description. The blind voting scheme using CAPTCHAs achieves the same goal as the basic blind voting scheme, but tries to increase the usability. There is no need to communicate any codes to the voters in advance to the election period using a second channel.

When the voter wants to cast his vote over the Internet, she has to follow the following procedure:

- The Internet voting application client authenticates the voter to the voting server using the national ID (eID).
- The Internet voting application client receives an image file in which the voter's personalized list is encoded as a dynamically generated CAPTCHA.
- The Internet voting application client displays an input box where the user enters a party number. The voter has to solve the CAPTCHA in order to get the correct party number.
- The Internet voting application client then displays a list of anonymous numbers that represent the candidates, and in a similar way, the write-in candidates can be added in the next step.
- When the user is done, the cleartext ballot is encrypted and digitally signed, and sent to the voting server for conversion using homomorphic encryption.

General Security Issues. The need for a secure second channel is eliminated. The Internet voting application client downloads the image from the server on-the-fly.

Eavesdropping Trojans. Eavesdropping is possible: the Trojan saves the image received from the server and the cast ballot. Both of them can be sent to the home base of the Trojan for analysis. This attack can be scaled if the attacker hires human CAPTCHA solvers, or implements a CAPTCHA solving algorithm.

Trojans Making Random Changes. If the Trojan can't solve the CAPTCHA, then the scheme is as secure as the basic blind voting scheme.

Trojans Voting for a Particular Party. If the Trojan can't solve the CAPTCHA, then the scheme is as secure as the basic blind voting scheme. If a Trojan can solve the CAPTCHA, then the scheme is no longer secure: the Trojan can cast ballots on the voter's behalf.

Other Security Issues. The defence against Tojans depends on their ability to solve CAPTCHAs efficiently. Microsoft's Live Mail service has been reported to be broken by spammers with a success rate of 30 % to 35 % [2], while Google's Gmail CAPTCHA was broken with a success rate of 20 % [3]. Notice however that the CAPTCHA challenge in this scheme is easier than guessing just a random sequence of letters like one often sees as protection on websites. Indead, the attacker knows in advance what will be the relevant words to look for in the CAPTCHA, i.e. the names of the parties. Just recognizing the first letter in a word may e.g. be enough to recognize the name of a particular party. It may therefore be necessary to change the CAPTCHA algorithm during the election period.

Usability. The user interface still looks awkward to people who are used to simple and intuitive user interfaces. Without modifications, people with disabilities may not be able to use the user interface either.

Open Source. The solution can be open source except for the CAPTCHA generation algorithms.

2.6 Tamper Indicating Open Voting Scheme

Description. This novel scheme [4] tries to solve the vote integrity problem, using a second and a third channel. For each voter, a polling card is generated containing control codes for each party, and these polling cards have to be distributed to the voters over a secure second channel. This may again be the Norwegian postal service, just like in the blind voting scheme, but may vary in other countries. The control codes should be sent back to the voter over a secure third channel, like e.g. over SMS. It is important that these codes aren't sent back over e-mail to the same computer the voter is voting from, since this will give Trojans the opportunity to manipulate the contents of the e-mail.

This time, the user has to do the following to cast his vote:

- The Internet voting application client authenticates the voter to the voting server using national ID (eID).
- The Internet voting application client receives a list of parties and candidates from the server; this list is not encrypted since it is the same for all voters in the same district.
- The Internet voting application client displays the list in a point-and-click interface.
- The voter makes his decision by clicking on the party name, adding additional votes for some candidates and adding write-ins too if he wants to.
- When the voter is done, the cleartext ballot is encrypted and digitally signed, and sent to voting server.
- The server then regenerates a control code from the encrypted ballot and sends it back to voter via the secure third channel (e.g. SMS) in a receipt. The message should contain instructions to verify the code against the code on the polling card.
- The voter checks whether the control code is the one for the party he voted for.

Eavesdropping Trojans. This scheme doesn't protect the confidentiality of the voter against Trojans. Since the ballot will still be present in plaintext in the memory of the voting application for at least a small amount of time, a Trojan could be used to post on a bulletin board who voted for which party.

Trojans Making Random Changes. A Trojan will not be able to make random changes to the ballot against the voter's will if the voter does verify the control codes he receives over the secure third channel against the one on his polling card.

Trojans Voting for a Particular Party. A Trojan will not be able to make targeted changes to the ballot against the voter's will either if the voter does verify the control codes he receives over the secure third channel against the one on his polling card.

Other Security Issues. Since this scheme is work in progress, more research needs to be done and will be done on its security before it will be implemented and used.

Usability. This scheme is slightly more complicated than the straightforward scheme due to the control codes and the use of a secure third channel.

Open Source. It is our goal that the security of this scheme lies in the private and secret keys used in the scheme, not in the mechanisms, such that it can be published as open source.

2.7 Discussion

Table 1 shows some rough estimates for the success rates Trojans will have for a number of specified attacks on the voting application. From this table, it is clear that if confidentiality on the client's computer is not an issue, the tamper indicating open voting scheme comes out best. If the confidentiality is an issue, then the blind voting scheme using codes is the best option, but as pointed out in subsection 2.3, the usability of this scheme is quite a challenge. If neither confidentiality nor integrity are an issue, the open voting scheme probably is the best choice.

Table 1. Estimates for the success rates of Trojans for the Open Voting Scheme Using Double Envelopes (OVS), Open Voting Scheme Using Double Envelopes, Open Source (OVS-OS), Blind Voting Scheme Using Codes (BVS-C), Blind Voting Scheme Using Symbols (BVS-S), Blind Voting Scheme Using CAPTCHAs (CAPTCHA) and the Tamper Indicating Open Voting Scheme (TIOVS) on eavesdropping, making a random change to the ballot, and making a change to the ballot in favor of a particular party. For simplicity, the figures are based on the assumption that 10 parties participate in a contest. For BVS-C and TIOVS, we assume the codes consist of five alphanumeric chartacters, i.e. digits and capital letters (minus captial O as to avoid confusion with the digit 0), thus allowing for 35^5 combinations. For BVS-S, the calculations are based on 100 different symbols being available. Finally, we assume the success rate to decipher a CAPTCHA to be somewhere between 70 % and 90 %.

Voting Scheme	Eavesdropping	Random alteration	Targeted alteration
OVS	90 – 100 %	90 – 100 %	90 – 100 %
OVS-OS	100 %	100 %	100 %
BVS-C	1.9×10^{-6} %	1.9×10^{-5} %	1.9×10^{-6} %
BVS-S	1 %	10 %	1 %
CAPTCHA	70 – 90 %	70 – 90 %	70 – 90 %
TIOVS	100 %	1.9×10^{-5} %	1.9×10^{-6} %

3 Trust

3.1 Introduction

The previous section discussed the security in the voting application. Many schemes were proposed and discussed, with their advantages and merits, but also their disadvantages and problems. In the end, a trade-off will have to be made if the Norwegian government wants to proceed with the *E-valg 2011* project and try out voting over the Internet.

Applying the right amount of security to a project may be a challenging task, but an even more challenging task for a system like *E-valg 2011* may be to gain trust for it. This trust can be situated on two levels: at the level of the Norwegian government, and in particular KRD, as the customer of the project, and at the level of the general public in Norway, as the user of the system to be developed.

The general public may again be divided into two groups: technical experts who can formulate qualified opinions about the security of the application and whether or not it can be trusted, and the non-experts who, to a certain degree, will rely on the technical experts to accept or reject the voting application.

Since elections are the cornerstone of any modern democracy, it is of paramount importance that the general public trusts the election system. If part of the election process is implemented using a voting system that the general public can't or won't trust, then there may be enormous consequences. The least consequence may be that the public chooses not the use the Internet voting system, in which case the money and time invested in the implemented system can be considered as a loss, and the project a failure. A far more serious consequence could be that the general public simply doesn't accept the legitimacy of the elected body, be it a local district council or a national parliament.

The strategy of the KRD to gain trust with the public for voting over the Internet seem to include three tactics: informing the general public about Internet voting and the concrete solution at hand as best as possible, a formal certification of the solution by a third party, and open sourcing the project so that those who are able to and want can verify for them selves that the system does what it is supposed to do in the correct way.

3.2 Information

The KRD, and in particular the *E-valg 2011* project group, has already created a website [5] where the general public can find information about what it is planning to do. It contains a lot of general information about voting over the Internet and electronic voting in general, including studies, reports, and links to other sites. There's also a FAQ section, which even before the end of the procurement process already includes questions about how privacy will be guaranteed, and whether the system should be trusted or not.

Experiences in other countries have shown that providing the general public with clear and understandable general information about the electronic voting system is essential for such projects to become a success. The Austrian case, discussed below, is an example of that. Of course, not everybody in the general public will have enough technical compentences to understand the inner workings of an Internet voting system, let alone the cryptographic aspects of it, but on a general level, people want the system to be explained to them in order to gain some trust in it. If for nothing else, one thing general information about the system definitely will show is whether the people in charge really know what they're doing or not.

3.3 Certification

Another, more formal way to gain trust in a system is to have it certified by a third party. For the customer, the Norwegian government represented by the KRD, this is probably the most realistic way to verify that the system has all the necessary security aspects, in addition to its own acceptance tests and a thorough follow-up of the project during all its phases.

The KRD has indicated that it wants the system to be certified through SERTIT, the public certification authority for IT security in Norway. However, the certification doesn't have to be in place for the 2011 pilot project, but rather for the 2013 parliamentary elections if the pilot project is a success and Internet voting to be rolled out over the whole country by then. One reason why KRD doesn't require a certification for the 2011 pilot project is to save time and money in the project, in addition to the fact that it is after all a pilot project that will be tried out on a rather small scale (though large enough to gain enough realistic experience). Another reason is probably that the project will be open sourced too, as discussed in the next section, which may actually put more pressure on the supplier than a formal certification would do. Finally, since a certification will be required for the final project to be delivered in 2013, and certification is a time consuming process, a lot of the preparations for a certification will have to be made during the development of the 2011 pilot project anyway.

3.4 Open Source

Finally, an additional way to let the general public gain more trust in a voting system is to publish its source code, e.g. by making it an open source project. The KRD has already decided and publicly announced [6] that it wants the *E-valg 2011* project to be implemented as an open source project. But what are the advantages and the problems with having such a project as open source?

One clear advantage would be that every voter has access to the source code of the project, and can, at least in theory, evaluate for himself whether or not he decides to trust the voting system. In practice, it can be expected that only a limited number of voters will do this exercise. Most probably, this will be a self-selected group of technical experts, and their opinion will have great influence on the general public's opinion about the system. In fact, if a large enough part of the system documentation is available in English, even people who do not have voting rights in Norway could participate in the exercise, and give their opinion too. The key point however is that a voter doesn't have to rely on the Norwegian government and/or a certification authority, and can review the code on his own, if he wants to.

One of the biggest disadvantages of having the project open sourced is of course that malicious hackers would have complete access to the source code too, and be able to carefully craft their attacks on the system. This doesn't make much of a difference for the part of the system that runs on the client side, since any code running there always can be grabbed from the client's memory and decompiled. Obfuscation of the source code could make this task a more difficult one, but considering that the advance voting period in Norway is three months long, it can be argued that the effect of obfuscation would be rather marginal. The problems associated to open sourcing the project therefore lies more on the server side: in an open source project malicious hackers have access to the source code on the server too, including a large part of the configuration necessary to run the system. Security by obscurity sometimes does have its merits, and this

is such an example. We think however that the disadvantages of having this project not open sourced outweight the disadvantages of having it open sourced, as we will discuss in the next section.

3.5 Discussion

Indeed, providing general information about the voting system and having it formally certified by a third party are two necessary conditions for the general public to gain any trust in it. Having the project open sourced seems to be more and more important too, especially where new voting systems are being introduced. The 2009 elections for the Austrian National Union of Students (*Österreichische Hochschülerschaft*, ÖH) are an example about how this demand from the general public made the project fail in the end.

What happened in Austria, is that even though the system had been formally certified, a popular movement called *Papierwahl.at*[3] was started against it. It is difficult to say how large the movement really was and how big its influence was, but it certainly was able to gain some attention in the media, and in the end only 0.9 % of the voter mass used the Internet application to cast its ballots [7]. One thing the Austrian government clearly seems to have done wrong is that it selected a rather small group of experts to have a look at parts of the source code for a limited amount of time. This move can be considered to be a compromise between having the project close sourced and open sourced. But instead of gaining some credit for the system, the government only received a lot of critique, both on the selection of which experts could join the code review –especially from people who would have liked to be included in the group– and the modalities on how the code review was organized [8]. It looks like the biggest looser of the election wasn't one of the participating parties, but the organizer of the election, the Austrian government.

4 Conclusion

Even though electronic voting seems to be very appealing with all its advantages, there are a lot of problems attached to it too. There have been many attempts in recent history to automate parts or all of the election process, not always with success. One country that has been able to run electronic elections without any major incidents is Estonia. In 2005, 2007 and 2009, voters had the possibility to vote over the Internet, and quite a few of them did so. It is the intent of our consortium, which includes the company that delivered the Estonian solution, to repeat this success in Norway, adapted to the specific Norwegian context.

At the moment of this writing, no decision has been made yet about which voting scheme is the most appropriate for the Norwegian context. The procurement process for the pilot project is due to result in the signing of a contract at the end of 2009.

[3] See http://papierwahl.at/ (accessed August 2009) for more information.

Acknowledgments

The authors would like to thank all their colleagues who have contributed so far to the *E-valg 2011* tender.

References

1. Okamoto, T.: Receipt-Free Electronic Voting Schemes for Large Scale Elections. In: Christianson, B., Lomas, M. (eds.) Security Protocols 1997. LNCS, vol. 1361, pp. 25–35. Springer, Heidelberg (1998)
2. Keizer, G.: Spammers' bot cracks Microsoft's CAPTCHA. Computerworld (February 2008), http://www.computerworld.com/s/article/9061558/Spammers_bot_cracks_Microsoft_s_CAPTCHA_ (accessed August 2009)
3. Prasad, S.: Google's CAPTCHA busted in recent spammer tactics. Websense.com (February 2008), http://securitylabs.websense.com/content/Blogs/2919.aspx (accessed August 2009)
4. Heiberg, S., Lipmaa, H., van Laenen, F.: On achieving e-vote integrity in the presence of malicious trojans (working title) (In preparation)
5. Ministry of Local Government and Regional Development: E-valg 2011-prosjektet (August 2008), http://www.regjeringen.no/nb/dep/krd/kampanjer/valg/elektroniskstemmegivning.html?id=437385 (accessed August 2009)
6. Ministry of Local Government and Regional Development: Forsøk med internettvalg i 2011 og bruk av åpen kildekode (July 2009), http://www.regjeringen.no/nb/dep/krd/kampanjer/valg/elektroniskstemmegivning/nytt-om-e-valg-2/nytt-om-e-valg/2009/forsok.html?id=570946 (accessed August 2009)
7. Kleijn, A.: Österreich: Nur 0,9 Prozent Wahlbeteiligung bei E-Voting. Heise Online (May 2009), http://www.heise.de/newsticker/Oesterreich-Nur-0-9-Prozent-Wahlbeteiligung-bei-E-Voting–/meldung/138303 (accessed August 2009)
8. Sokolov, D.A.J.: E-voting ist in Österreich nicht unbedingt geheim. c't Magazin (May 2009), http://www.heise.de/ct/E-Voting-ist-in-Oesterreich-nicht-unbedingt-geheim--/artikel/138049 (accessed August 2009)

Advanced SIM Capabilities Supporting Trust-Based Applications

Thomas Vilarinho[1], Kjetil Haslum[2], and Josef Noll[3]

[1] Department of Telematics at NTNU and Telenor R&I, Trondheim
tcarlyle@gmail.com
[2] Telenor R&I, Trondheim
kjetil.haslum@telenor.com
[3] University of Oslo/UNIK, Kjeller
josef@unik.no

Abstract. The SIM cards are going through several new enhancements both in the underlying hardware and its capabilities. They are becoming secure wireless networked devices containing embedded sensors. This paper assess how this new capabilities together with the pervasiveness and security of the SIM card can support the development and design of trust-based applications. Moreover, we present a specific use-case around a seamless trust builder for social networks, which makes use of sensed inputs towards building hard contextual evidences to trust relations. We conclude with the description of the challenges of building this evidence based trust-builder and the necessary steps to going from the prototype we developed to a real application which may accurately describe trust relations.

Keywords: SIM cards, trust, networked embedded systems, pervasive computing, Sun SPOT, social networks, context-awareness.

1 Introduction

It is unanimous that the mobile phone is the most popular personal pervasive device so far. The strong presence of mobile phones, together with the development of new interfaces and sensors, has pushed several applications to be developed on this platform. However, the mobile phone itself is not considered a platform truly secure as it seldom has memory access protection or physical tampering sensors. On the other hand all mobile devices represented by the Global System for Mobile communication (GSM), which corresponds to more than 80% of the mobiles[1], have a security element represented as the Subscriber Identity Module (SIM) card.

The SIM, as a smart card, corresponds to a well trusted and tamper-proof device. Smart cards are trusted enough to play key roles in highly secure business cases such as banking, key management, identification and authentication. Using the SIM instead of the Trusted Platform Module (TPM) has the advantage of

[1] http://www.gsmworld.com/newsroom/market-data/market_data_summary.htm

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 223–238, 2009.

having a physical element which the user can remove. Thus it offers the possibility for identity mobility, as the identity of the device (through the SIM) can be moved from one device to another device.

Besides the access to all features from the mobile or from any other device to which the SIM card is connected, new physical and logical interfaces are becoming available for the SIM. Standard and non-standard wired and wireless interfaces and sensors are being integrated with the SIM. In addition to that, we are starting to see more and more cases of multi-application cards. These new capabilities being developed around the SIM, allied with its pervasiveness and security, enable it to play different new roles, such as a trust component for federated identities, a secure platform for the Internet-of-things, a seamless provider of contextual evidences, or a mix of those.

On the other hand the phenomenon of online social networks (OSN) has also reached an enormous number of users. Their popularity is so big that about 20% of the Internet page views currently are from the social networks MySpaces and Facebook, where this corresponds to half of the views between the top 10 most popular domains, [1]. Facebook, in 2009, had more than 200 million active users where half of them access it at least once a day, while MySpace reported more than 110 million active users in 2008. Those two Social Networks together have more active users than the whole population of the United States, the third most populated country in the world.

However, the relations established in those social networks are not representative enough for serving as a base to attribute the trust between users that have a relationship defined there. Breslin and Decker point out that in many OSNs people connect to each other for only boosting their number of connections [1]. Moreover, several users feel compelled to accept friendship invitations despite they would not do it in the real life [2]. A survey done with some users of the Orkut social network showed that about one fourth of the connections that those users have done was due to a feeling of an obligation, as users preferred to add an unwanted friend instead of possibly offending the person [3]. Due to the mentioned problems and the fact that those OSN relations often do not carry attributes that characterize them, it may be misleading to assume that one user trust the other, in a general or restricted context, based on those virtual relations.

We see that the new advanced features on the SIM card can offer a solution to the lack of trust in the user's connection in social networks. This future SIM can counter impersonification through mutual authentication mechanisms, but it can also seamless sense the real life interactions between the users and offer real hard evidences towards the description of the users' trust relations.

In this paper we asses some situations that could benefit from this new capabilities of the SIM, and describe the case of a transparent trust builder. This trust builder makes use of the identities stored in the SIM card and the context information acquired during the contact between two SIMs in order to characterize the trust relation between the two SIM card owners. We have developed a small prototype using Sun SPOTs emulating this future SIM card in the seamless trust building scenario. Despite using a simple trust logic and limited contextual

information we achieve promising results and indications of what other factors could be modeled in order to reach a more accurate trust builder.

The rest of this paper is organized as follows. Section 2, we describe the state-of-art of the SIM cards and its applications on trust building; afterwards, we present the related work towards using the SIM to build trust and around categorizing relations. Then, in section 4, we explain the seamless trust builder and the experiments and results we achieved. In the final part of the paper we conclude with presenting directions of future work.

2 The State of the Art in SIM Card Technology

Thanks to the high rate of evolution in the telecommunications, SIM cards have leaded the advances of smart card functions. Their main function has been to prove the authenticity of the mobile device in respect to the network. But as theirs capabilities were expanded, they became the secure element for several applications, acting as identity and profiling devices for the user, and as a secure channel between theirs applications and the mobile phones.

One of the driving factors to the continuous raise of the importance of the SIM card is its security characteristics. Both hardware and software passes through rigorous development process, enforced by auditions and clear standards. Thanks to that process, the smart card industry manages to be ahead of the attacks that can be performed at lower cost than the value of the data secured in the card [4, 5]. Moreover, the smart cards offer a multi-application architecture where a firewall is established between the applications securing their execution and the data access. This is assured by the implementation of the Global Platform (GP) standards [6], and the concepts of the Card Manager and Security Domains.

SIM card applications such as JavaCard applets or SIM browsing, as Wireless Internet Browser (WIB) and S@T, use the SIM as a trusted platform in order to exchange messages via Over-the-Air (OTA) platforms. Those applications can be remotely loaded and managed, exchange messages and use cryptographic primitives through the security domains implemented in the SIM and standardized by the Global Platform specifications.

Those applications depend on the SIM Application Toolkit (SAT) to access the mobile phone capabilities, such as intercepting phone calls, sending messages, GUI methods, sensors and communication interfaces, such as Bluetooth and General Packet Radio Service (GPRS). The SAT specifies few interactions commands with the mobile, besides the management of logical channels between SIM and handset. However, in order to exchange large portions of data through those logical channels, both mobile phone and SIM need to implement a higher level data transfer protocol such as the Bearer Independent Protocol (BIP) or native TCP/IP protocol. So far, few handsets implement BIP and the native implementation of TCP/IP in the SIM cards has just been recently standardized by the European Telecommunications Standards Institute (ETSI) in [7].

J2ME applications running in the mobile can communicate with the SIM through the Security and Trust Services API for J2ME (SATSA); JSR 177. This

API allows the J2ME Midlets to exchange Application Protocol Data Units (APDUs) with the SIM. By that, the Midlet can use the SIM to store keys and perform cryptographic operations such as digital signatures, encryption and authentication, as explored in [8]. A high level communication protocol between handset and SIM is the recent SIM Card Web Services (SCWS), specified in [9]. The SCWS enables the application in the SIM Card to be accessed through the terminal's browser, through BIP or TCP/IP, and to be managed through OTA.

A great breakthrough in the SIM cards has been the expansion of its communicating capabilities. The communication to the local reader has been improved from the 9.6 Kbits/s rate of the T=0 and T=1 protocols to around 8-12 Mb/s from the USB protocol. This is actually one of the driving factors for the development of higher bandwidth protocols in the SIM, such as the SCWS and the implementation of native TCP/IP. Besides the advances in the physical communication, recent releases from SIM cards producers such as Samsung and SanDisk have managed to deploy smart cards with powerful 32-bit processors and more than 1GB of memory.

Other interfaces are emerging around the SIM as well. The SIM pin that was used in the past for the programming voltage has become the connecting pin to the Single Wire Protocol (SWP), the physical link to a Near Field Communication (NFC) module standardized in [10, 11]. Moreover, Smart Card manufacturers and operators have announced the integration of accelerometer[12], GPS[13], IEEE 802.15.4 [14] and IEEE 802.11[15] interfaces directly on the SIM. All those interfaces grant to the SIM a high level of connectivity and context awareness independent of the terminal capabilities. Furthermore, ETSI has been working on standards to specify SIM cards that have more robust physical characteristics, targeting the machine-to-machine (M2M) market, as evidenced in the 3GPP M2M feasibility study [16].

In order to comply with those hardware advances, the JavaCard, the most penetrated smart card platform, has just seen a major improvement in its specifications. The new JavaCard 3.0 release supports more classes and features of the Java SE, multithreading, nested transactions, annotations and it adds an unified naming scheme for both applications and theirs resources [17].

There are several examples of applications and projects using the SIM card we have nowadays as a trust element. Some examples are Digital Rights Management (DRM), Mobile-banking, one-time password (OTP) solutions, Biometric verification, unified authentication, etc. The current potential of the SIM towards a local identity manager is further discussed in [18]. This identity management is especially improved with the new hardware changes in the SIM, and it has triggered the specification of an Identity Management (IdM) Framework by the GSMA [19].

This IdM framework breaks the definition of the Identity Provider in Authentication Provider and Identity Attribute Provider. The Authentication Provider is responsible for validating the user's credentials and providing him authentication assertions. In contrast, the Identity Attribute Provider facilitates sharing user attributes to trusted parties.

There is a great speculation over this attribute sharing capability, as it could offer extremely valuable information for content customization since users seem to be willing more and more to share personal information, as observed in [20, 21]. Once the SIM card starts to aggregate the mentioned sensors and communication interfaces, it can fetch real time context attributes that could be eventually shared through this Identity Attribute Provider, enhancing even more the customization potential for Value Added Services. In fact, as a pervasive device which is almost always with the user, it becomes the perfect platform for this real-time context sensing.

The diversity of sensors and interfaces allows the deployment of a context characterization and quantification framework just as described in [22] and [23], where all the inputs are combined to generate a context value with its context quality parameters such as: freshness, accuracy, precision, reliability and granularity [24]. Or this diversity of sensors could lead to optimize the context acquisition in terms of use of sensors and consumed battery based on the desired threshold of context quality parameters.

This context and its attributes can be applied as hard evidences towards the user or object, in the M2M case, when building trust around that context. This hard evidence capability can be endorsed by an authentication by the user or physical mechanisms on the SIM to detect that it hasn't been physically tampered, as the logical isolation is already provided by the SIM operational system and its security domains.

The potential of context-awareness is enormous. The wireless interfaces and NFC enables the SIM to sense the surrounding devices. This, together with the fact that the SIM card carries at least one identity (the Mobile Subscriber ISDN Number), enables to sense the relation between people: how often do they meet, how long do their meetings last, and which activities do they perform together. Moreover, those SIM could sense the surrounding objects that share their attributes, not only RF-ID tags but possibly other wireless devices that offer web services to reachable devices. And, by that acquire more information about the surrounding environment.

3 Related Work

The SIM card already plays a role for building trust as it is used as the main component to the GSM Authentication in the GSM Networks. It corresponds to a complex case of Identity Management (IdM) and policy-based trust, as the subscriber can seamless roam in different networks as long as there is a roaming agreement between the operators, somehow similar to Single-Sign-On (SSO) case based on a federation agreement between Identity Providers. National ID cases, such as the FINEID (Finish ID), Austrian ID and MyKad (Malaysian ID) for example, have successfully deployed IdM solutions hosted on the SIM Card, thanks to its multiapplication and applet firewalling capabilities.

Despite the cases of building trust based on shared keys in the SIM and in the authority that issues those keys in order to perform identity management,

mobile banking, authentication or controlling data access; we could not find any approach in the scientific literature towards sensing context with the SIM in order to characterize relations between people as in the seamless trust builder we have designed. In [25], Mayes et al discuss the potential of high density smart cards, but the article mainly focus on the high capacity and the increase of the speed in the physical interface of the SIM, not much on its potential as personal trusted context-aware platform. A framework for treating information from sensors connected to the mobile in order to generate a high level and quantified is presented in [22]. But, [22] does not go into the details of the usage of the context information.

In the other hand, [26] and [27] tries to seamless characterize trust relations between individuals, but based on theirs distance in OSNs. It is a valid approach as long as those relations in those social networks carry attributes that make them meaningful. The characterization of those relationships with the focus on the mutual reliance between the users is something that the seamless trust builder aims to do.

4 The Seamless Trust Builder

As mentioned in the introduction, the relationships in online social networks (OSNs) do not truly characterize trust relationships between its users. The reason for that is the lack of attributed information on the OSN links to serve as base for representing how much and in which kind of situations one user would trust the other. As trust, we consider how much the user could rely on the other's willingness and capability to act as expected on situations like giving advices, providing a service, etc. In fact, some social networks, such as Facebook, are adding mechanisms to arrange friends on group categories. However, defining those groups in the current OSN demands a lot of engagement from the users. Thus, the seamless trust builder could sense evidences in the user relations and transparently categorize the relationship between the users, giving them real meaning and suitable for inferring the trust between users.

4.1 Contextual Evidences

Considering the whole conjunct of interfaces and sensors that have been so far announced we identified a few context and sensors that could be retrieved by this future SIM, as summarized in Table 1.

Theoretically it would even be possible to have access to other sensors in the mobile such as: camera, microphone, light sensors (commonly present for the camera) and temperature sensors (commonly present for battery monitoring), as noticed in [28]. However the temperature and light sensors are usually not open to be used by 3rd party applications and the camera and microphone may be too invasive or demand too much processing and storage.

Still, with the sensors listed in the table we can already acquire a great amount of information, especially if we consider the access to information stored

Table 1. Sensors and contexts available for the SIM trust builder

Context	Sensing Interface	Availability
Location	GPS, CellId; or NFC, IEEE 802.15.4 and IEEE 802.11 for relative position	Embedded in either standard or prototype SIMs
Environment, or surrounding devices	NFC, IEEE 802.15.4 and IEEE 802.11	Embedded in either standard or prototype SIMs
Motion	Accelerometer or based on location over time	Embedded in either standard or prototype SIMs
Activity	Accelerometer to some extent or application in the mobile or SIM	SIM can host applications and communicate with apps on the phone
User Profile	User information in databases linked to his identities	SIM can hold several identities

in databases linked with the user's identity. Thanks to the facilitation to publish content provided by the Web 2.0, nowadays users leave several digital footprints in the web. Gathering those attributes (such as the home address of the user, his profession, interests, family tree, etc) can offer almost unlimited possible inputs to enrich the description and contextualization of his profile or a trust relation held with other users. It would be possible to infer if the users share common interests; have distinct or similar points of view; are family related; etc.

4.2 The seamless trust builder

In fact the seamless trust builder uses the SIM context-awareness at two different stages. First, to detect that two users are close to each other, as we infer that due to this proximity they share some kind of relation or interaction. Then, it retrieves information about the context where both users are, gathering the necessary inputs to describe the interaction between the users. Then, based on this mutual interaction, it attributes a trust value between them.

When reasoning about a situation where a user needs to assess its reliance in the other agent, he can take into account a recommended trust (based on the recommendation of trusted sources), a historical trust (based on previous similar situations) or his dispositional trust towards that agent. This dispositional trust is what the literature [29] refers as default trust behavior of the trusting agent, his basic tendency to rely on the other agent. And, it is the dispotistional trust that the trust builder aims to give a value to.

Native radio interfaces such as NFC, IEEE 802.15.4 and IEEE 802.11 would be enough to sense other future-SIMs in the radio range, as a proximity sensor. But, in order to protect the user's privacy, he must be able to decide to broadcast or not his identity. Moreover, if he does, he should be able to choose to identify himself through a pseudonym instead of his real name, or possibly have his pseudonym or identity dynamically chosen based on the context. A mutual

authentication mechanism, such as the Transport Layer Security (TLS) handshake, could be first required in order that the SIMs exchange more information, as their pseudonym or even profile attributes. By that, it would be possible to correlate the context in the situation just in case the users have some trust level between themselves or between a common trusted authority.

Once the interaction is detected, it is time to sense the context information so it can be categorized and serve as an input to generate a trust value. Based on the previous section, we have identified the following inputs for describing the interactions: duration, timestamp, location, profiling, and the user's activity. The duration can be calculated by measuring the time elapsed since the moment where the users get in contact and separate from each other. The timestamp can be retrieved by the phone or network clock, while for the location there are several possible sensors. Some location sensor could provide an absolute location, based on a numeric coordinate cartographic system, or a symbolic location, such as in a shopping center, at home, etc. The profile attributes could be fetched through the user's identity and connection to the Identity Attribute Server of that identity.

The activity of the user is much harder to be categorized. First of all, because the user may be engaged in more than one activity simultaneously such as walking while talking in the mobile and watching an outdoor in the street. It is not so hard to sense the activities which are based on the mobile device, such as using a mobile application or talking on the phone, as this could be sensed by the SIM through the interfaces towards the mobile such as SAT, SCWS, JSR-177, etc. But for other activities, which are the ones the user spend most of his time, sensing them is much more complex. This complexity includes the difficulty to sense those activities and to describe them.

For the sensing for example, the accelerometer can track the user motion, but we would need to recognize the activity based on the movement pattern. As an example, a study done by Karantonis et al [30] tries to identify a few distinct user activities with focus on elderly monitoring. They try to recognize situations such as if the user is standing up, sitting, falling and walking. Although, they managed to detect with good accuracy the changes between activity and rest, the detection of the cases where the person was walking was not accurate.

On the other hand other contextual inputs can help to deduce the user activity. As for example, if it is sensed that he is moving, but inside a football field, there are more chances that he is playing football. However, it is common to find exceptions in most of the cases, as for example the goalkeeper would have very different moving patterns from an attacker, although both of them are playing football. This is mainly a problem on describing the activity and it can be sometimes countered by limiting the scope of activity characterization. The activity could be limited for example just as if the user is busy or idle.

Based on the mentioned inputs of activity, location, timestamp and duration sensing, it is necessary to deduce a trust value from it. As pointed out in [31], the majority of the current reputation systems do not differentiate between general trust and contextual trust although it is an important issue. A user may trust

another about work related topics but not on personal issues. For our seamless approach, we can benefit from the sensor to help detecting the context. This motivated us to generate not a single dispositional trust value, but a few different contextual dispositional trust values. Those values could be later used to create a more advanced user social network representation or to automatize or improve decisions that rely on another user.

Most of the challenges towards attributing the right quantization of some contexts and the calculation of accurate trust values are closely related to pattern recognition whose root are in the sociology, psychology and cognitive sciences. However, in this paper we focus on the sensing capabilities and, consequently, we don't focus on the pattern analysis. We use a simple model and simple scenario in order to build a proof-of-concept application showing the potential of the SIM to become a passive trust builder in the future due to its sensing capabilities.

4.3 Implementation

As this paper explores the capabilities that are being developed on the SIM but are not yet present in a product available for developers, we decided to do our implementation in a Sun SPOT (Sun Small Programmable Object Technology).

The Sun SPOTs are small, wireless devices embedded with 3 different sensors (temperature, light and an accelerometer) besides I/O general purposes pins that enable the connection of additional sensors. The Sun SPOT hosts a small-footprint J2ME java virtual machine running directly over the hardware. The hardware consists of a 180 MHz 32 bit ARM920T core processor with 512K RAM and a 2.4 GHz IEEE 802.15.4 antenna. There are two types of Sun SPOTs: the free-range SPOTs which have all the sensors; and the basestation SPOT which does not have sensors (just the IEEE 802.15.4 interface) but can connect to a PC by USB port and serve as a gateway between the other SPOTs and the PC.

The Table 2 lists the pros and cons of using the Sun SPOT for emulating this future SIM.

Due to the restrictions of the Sun SPOT platform it was not possible to gather data about the user's activity based on the applications running on the device as it is not a phone nor contains applications similar to the ones used on the mobile phones. Theoretically it would be possible to infer some physical activities based on pattern recognition around the accelerometer. But based on

Table 2. Restrictions in using the Sun SPOT for emulating an enhanced SIM

Advantages	Disadvantages
Virtual Machine and API similar to Javacard 3.0	No GUI on the device itself
Accelerometer, temperature and light sensors embedded	Somehow limited cryptographic support
802.15.4 radio communication interface	No absolute location sensor
RSSI radio feedback can sense proximity	Hard to use for a real scenario simulation
Portable and Mobile	It is not a phone

[30] and in experiments we have done with it and the Sun SPOT Telemetry Demo, we could not be able to recognize well activities patterns, as depending on the orientation of the SPOT and the type of physical contact applied to it, the patterns would change. Having stated that, we used the accelerometer to detect a change of state of the user when he starts to move and stops to move. This change of state was employed to optimize the usage of the radio interface as a SPOT would only broadcast sensing packages when it moves.

We used the IEEE 802.15.4 interface for sensing the proximity of two or more SPOTs. We periodically exchanged the Received Signal Strength Indication (RSSI) measurements packets until the connection is broken due to the fact that one of the SPOTs leaves the radio range. Although the radiation power is proportional to the distance, it suffers great variations from interferences. As a result, some tests we have performed signalized more less the same RSSI values for SPOTs in the same room or in rooms separated by walls. Consequently, we decided to use the proximity contextual information only to detect if the SPOTs could sense each other, meaning that their owners were together engaged in a certain interaction further modeled based on their location.

We used the IEEE 802.15.4 packets for sensing the location as well. The location information has been retrieved from location broadcast datagrams emitted by a basestation acting as a location information provider. This was done to emulate the fact that the SIM card would be able to retrieve the location from a mapped Cell-ID, GPS or even from a wireless hotspot. It is in fact feasible to map the hotspots and acquire significant position information, as done by Skyhook or Google Gears Geolocation API. The location packets provide a contextual location that in our scenario was divided in home, outdoor, my office and office's corridor. Nevertheless, the real application should handle more types of contextual locations and possibly be able to convert absolute geographical locations into contextual ones. An illustration of the interaction environment with the 2 free-range SPOTs and the basestation is shown in Figure 1:

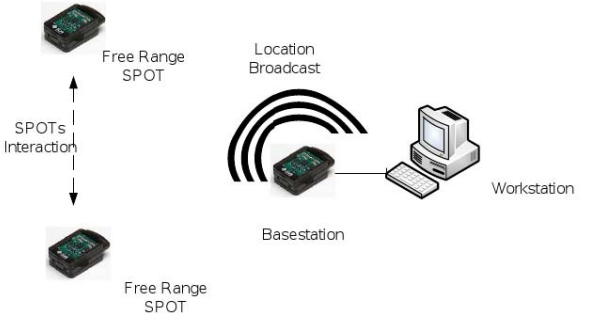


Fig. 1. Communication between the agents in the Sun SPOT implementation

We made some tests on fetching attributes linked to an identity managed by an Identity provider, as we personalized the SPOTs with aliases from the social network Last.fm and we used its REST API to retrieve some data. We

successfully fetched the favorite artists of the users and their musical compatibility through HTTP requests issued from the SPOTs and routed by a SPOT basestation. However, we opted for not using the attribute values from linked data once most of the work would be related to data mining and crawling, and it would make the trust calculation logic quite complex. Still, we recognize that due to the richness of digital footprints, this kind of information should be taken into account in a future application.

Being our sensing restrictions set and our goal focused on a proof-of-concept application on the sensing potential, we decided to categorize a trust value for each interaction based on its location and duration. Afterwards, the dispositional trust would be calculated taking into account all the interactions between the users.

For attributing the trust value for each interaction, we first quantify the duration as {passby, short, medium, long} and the location as mentioned before. Based on those values we attribute trust values to three different contextual trusts: professional, private and public as show in the Table 3.

Table 3. Trust Matrix

Location	Duration	Equivalent situation	Prof. Trust	Pub. Trust	Priv. Trust
Home	PASSBY	Someone passes by, no spoken contact	0	1	0
	SHORT	Possibly a home delivery or small talk at the door	0	1	1
	MEDIUM	Someone that is invited to come in	0	2	3
	LARGE	Other people living at home or passing the night	0	5	5
My Office	PASSBY	Someone passes by, no spoken contact	1	0	0
	SHORT	Asking for work-related information, small chat	2	1	0
	MEDIUM	Meeting	4	2	0
	LARGE	Working together in the same room	5	3	2
Offices Corridor	PASSBY	"Cross each other, no spoken contact"	1	0	0
	SHORT	Stopping and chatting in the corridor	1	2	0
	MEDIUM	Activity being performed as having lunch together	3	3	2
	LARGE	NOT DEFINED	-	-	-
Outdoor	PASSBY	Cross each other, no spoken contact	0	0	0
	SHORT	Crossing each other leading to small chat	0	1	0
	MEDIUM	Enrolled in an activity together: cinema, shopping, sport, etc	0	2	1
	LARGE	Enrolled in a daily activity together: travelling, camping, etc	0	4	3

The private context symbolizes relationships on the private level (such as family and friends). The professional context represents relationships related to the person's profession or work. And finally, the public trust context tries to represent people that the user knows, shares some interest but wouldn't necessarily

Table 4. Trust Scale

Trust Value	Professional	Private	Public
1	They are from the same company	Somehow privately related	Have met in public contexts
2	They know each other in the work environment	Acquaintance	Know each other from public activities
3	Work in the same department	Close Friend	Share a common interest
5	Work in the same room	Family	Part of a common interest group or organization

share facts about his personal life. The range of the trust scale used can be seen in Table 4, where each trust value is linked to some meaning, some equivalence that motivated us to quantify it. The 0 is not represented but it means that the agents haven't met. In this model, we do not represent distrust.

In order to assign the value of the contextual dispositional trust between the users at an instant, we put all the values from the previous interactions and their timestamp into a deterministic formula, so that the time degradation of the trust can be represented as well. Instead of using the time elapsed since the interaction as a value measured in time units, we quantify it in three categories recent, past, old. Where recent corresponds to interactions not older than a month; past for the ones in between a month and a year; and old for interactions older than a year. The final formula used is represented by Eq. 1:

$$T_{context} = \frac{0.7}{n'_{recent}} \times \left(\sum_{i=1}^{n'_{recent}} T'_{recent_i} \right) + \frac{0.25}{n'_{past}} \times \left(\sum_{i=1}^{n'_{past}} T'_{past_i} \right) + \frac{0.05}{n'_{old}} \times \left(\sum_{i=1}^{n'_{old}} T'_{old_i} \right) \tag{1}$$

where

$$T'_{time_i} = \{\ T_{time_i} \mid T_{time_i} \neq 0\}$$
$$n'_{time_i} = \left| T'_{time_i} \right|$$

and the time represents the time categories *old*, *past*, *recent*, and T_{time_i} represents each one of the trust values of that time frame category.

4.4 Experiments

The experiments actually started with tests towards recognizing user activity patterns with the accelerometer, inferring the SPOTs proximity through RSSI measures and checking the range of the IEEE 802.15.4 antenna.

After we identified the context information we could use, we modeled the trust formula mentioned before and we decided to try it in a small fictional scenario where a relationship between two users is emulated. We created a script describing their relationship, and the interactions that would result of their relationship. Based on that, we decided to individually test the trust model, and then, test the sensing acquisition and input of data in the database.

We considered as the emulation scenario the relationship of two friends that work at the same company but at different departments. Since they work at different departments, they have distinct routines. At work, they mainly meet once a week for having lunch and twice a week while having a small chat at one's office room. Their friendship goes beyond the office routine as once per month they either do an outdoor activity (such as going to the cinema, jogging, etc) or they have dinner at each other's place.

We emulated this routine for the equivalent of two months of the scenario and we noticed that the values of professional trust between the users was in between "knowing each other from work" and "working in the same department", while privately they were close to "acquaintance" and publicly to "know each other". While the professional and private values seemed reasonable, the public one looked somehow underestimated. There was an improvement, although not that significant, of the trust with the insertion of "a common activity outside" and "a dinner at home" between the friends. We also introduced an event corresponding to a trip together for 3 days, and, at this point, the private trust has been increased to a level closer to "close friend", but the public one was still far from "share common interest" level. At last, we simulated a month without contact between the agents, representing a vacation period. After that one month period, the trust events moved to the past category, and their weight in the trust value severely decreased. Nevertheless, they were quickly recovered when the users started again their weekly routine at work together.

4.5 Evaluation

The simple trust inference model shows decent results for the scenario emulated as it converged to reasonable values of professional and private trust between the users and showed both resistance to single events and degradation of the trust values in the case the users lose contact. However this degradation, the event resistance and the accuracy of the values should be reviewed by experts from the sociology and psychology field as they probably will be able to draw more relevant conclusions and propose unbiased adjusts for the logic and quantifications.

Still, we managed to identify a few aspects that could be improved in the model we have used. For example, the mentioned resistance to new inputs is at some point valid as it limits the trust variation from a biased or wrong input. On the other hand it makes the trust value too much dependent on the user's routine. The ideal solution to address this would be to weight the different activities in comparison with the frequency associated with that specific activity, even if this may require much more work characterizing the activities and defining the right

weights. In fact the number of events could be a parameter to enhance the trust values, so additional events always contribute positively to the trust value.

We considered modifying some values, especially for the public trust results, on the trust matrix. However, we see that some occasions would assign a trust that does not really exist, i.e. where both users are present at the same place and in the same range but not really establishing trust (as if two people take the same plane for example). For this problem we believe that a further analysis on the activity context could give better inputs to find suitable values for the public trust of the matrix.

Another point that we observed in our experiments is that it would be beneficial to expand the contact from only presence-based to include also other communication formats such as phone calls, e-mails, etc. This would be necessary in order to accurately represent cases such as when the agents separate from each other but still keep contact. It is important to capture the real life interactions, but the virtual ones should be considered as well, possibly with a smaller weight.

In what concerns the platform itself, the Sun SPOTs revealed to be an adequate platform for representing a full featured future SIM. It does lack a real GUI and it lacks the connections with a mobile and a real location sensor. However, its java machine is very powerful and easy to code for it; it is mobile, has a great range of sensors and it is possible to attach other pieces of hardware to it.

We noticed also that the application must define how to consider small interaction interruptions. For example, if two agents are working together in the same office, their interaction would suffer brief interruptions when each one goes separately to the coffee machine or just merely leave the range for a while. The definition of the interval corresponding to the break of the connection should depend on the context of the interaction, the range of the sensing capabilities of the agents and the model itself.

5 Conclusions and Future Work

This work presented an approach towards using the new advanced sensing and communicating capabilities of the SIM card in order to support trust applications and specifically trust relations in social networks. We established functionalities of future SIM cards such as motion and location detection by emulating them on the Sun SPOTs.

Several context sensing possibilities were identified for modeling the trust relation between two entities based on hard evidences sensed during their interactions. Moreover, we used some of those contexts in a simple model to prove this seamless trust builder concept. We emulated a routine of co-workers spanning an equivalent of two months, resulting in values of professional trust between the users as an intermediate of “knowing each other from work” and “working in the same department”. Their private trust value was close to “acquaintance” and their public trust value “know each other”.

The experiment has shown that attributing dispositional trust values between users based on sensing the context between their interactions is feasible. First results for this sensor-supported trust model look promising. In order to accurately represent the dispositional trust between users, it is necessary to further adjust the input values for our trust model by characterization and quantization of the inputs and outputs with experts of cognitive sciences. Moreover it is advisable to deploy a broader experiment with several distinct cases and preferably with real people.

References

[1] Breslin, J., Decker, S.: The future of social networks on the internet: The need for semantics. IEEE Internet Computing 11(6), 86–90 (2007)

[2] Beattie, R.: Linking out after two years of linked in (2005), http://www.russellbeattie.com/notebook/1008411.html (Last accessed: 31/05/09)

[3] Ann Golbeck, J.: Computing and applying trust in web-based social networks. PhD thesis, University of Maryland, College Park (2005)

[4] Renaudin, M., Bouesse, F., Proust, P., Tual, J.P., Sourgen, L., Germain, F.: High security smartcards. In: DATE 2004: Proceedings of the conference on Design, automation and test in Europe, p. 10228. IEEE Computer Society, Washington (2004)

[5] Markantonakis, C., Mayes, K., Tunstall, M., Sauveron, D., Piper, F.: Smart card security. In: Nedjah, N., Abraham, A., de Macedo Mourelle, L. (eds.) Computational Intelligence in Information Assurance and Security. Studies in Computational Intelligence, vol. 57, pp. 201–233. Springer, Heidelberg (2007)

[6] GlobalPlatform Inc.: GlobalPlatform, Card Specification Version 2.2 (March 2006)

[7] ETSI: ETSI TS 102 483: Smart cards; UICC-Terminal interface; Internet Protocol connectivity between UICC and terminal V8.1.0 (April 2009)

[8] Eisl, F.: Smart card security services for an open application environment used in mobile phones (June 2004)

[9] OMA: Smart Card Web Service Standard Version 1.1 (2009)

[10] ETSI: ETSI TS 102 613: Smart Cards; UICC - Contactless Front-end (CLF) Interface; Part 1: Physical and data link layer characteristics V7.5.0 (April 2009)

[11] ETSI: ETSI TS 102 622: Smart cards; Smart Cards; UICC - Contactless Front-end (CLF) Interface; Host Controller Interface (HCI) V7.4.0 (April 2009)

[12] PRNewswire.com: Oberthur technologies announces simsense - the first motion detection sim card (2009), http://news.prnewswire.com/ViewContent.aspx?ACCT=109&STORY=/www/story/02-16-2009/0004972701&EDATE= (Last accessed: 31/05/09)

[13] Cellular-news: Sagem to embed gps receiver into sim cards (2008), http://www.cellular-news.com/story/34691.php (Last accessed: 31/05/2009)

[14] Brede, S.: ETSI Workshop presentation June 4-5, 2008 A couple of M2M activities: WLANSIM a wireless IP networked UICC. Telenor R & I (2008)

[15] Turolla, M., Alessio, E.: Presentation ZSIM enabling innovative services to improve quality of life. Telecom Italia – Innovation & Engineering (2006)

[16] 3GPP: ETSI TR 133 980: Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Liberty Alliance and 3GPP security interworking; Interworking of Liberty Alliance Identity Federation Framework (ID-FF), Identity Web Services Framework (ID-WSF) and Generic Authentication Architecture (GAA) V7.6.0 (October 2007)
[17] Sun Microsystems, Inc.: Runtime Environment Specification: Java (TM) Platform, Version 3.0, Connected Edition (March 2008)
[18] Johannessen, T.: Identity management in general and with attention to mobile gsm-based systems. Telektronikk, 31–51 (2007)
[19] GSM Association.: Identity Management Framework Document V1.1 (2008)
[20] Stutzman, F.: An evaluation of identity-sharing behavior in social network communities. Journal of the International Digital Media and Arts Association (2006)
[21] Fogel, J., Nehmad, E.: Internet social network communities: Risk taking, trust, and privacy concerns. Computers in Human Behavior 25(1), 153–160 (2009)
[22] Korpipää, P., Mäntyjärvi, J., Kela, J., Keränen, H., Malm, E.J.: Managing context information in mobile devices. IEEE Pervasive Computing 2(3), 42–51 (2003)
[23] Schmidt, A., Laerhoven, K.V.: How to build smart appliances. IEEE Personal Communications 8, 66–71 (2001)
[24] Hristova, A.: Conceptualization and design of a context-aware platform for user-centric applications (June 2008)
[25] Mayes, K.E., Markantonakis, K.: On the potential of high density smart cards. Information Security Technical Report 11(3), 147–153 (2006)
[26] Katz, Y., Golbeck, J.: Using social network-based trust for default reasoning on the web
[27] Taherian, M., Amini, M., Jalili, R.: Trust inference in web-based social networks using resistive networks. In: ICIW 2008: Proceedings of the 2008 Third International Conference on Internet and Web Applications and Services, pp. 233–238. IEEE Computer Society, Washington (2008)
[28] Bražinskas, R.: Towards context awareness using mobile sensors (2008)
[29] Chang, E., Dillon, T., Hussain, F.: Trust and reputation for service-oriented environments: technologies for building business intelligence and consumer confidence. Wiley, Chichester (2006)
[30] Karantonis, D.M., Narayanan, M.R., Mathie, M., Lovell, N.H., Celler, B.G.: Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring. IEEE Transactions on Information Technology in Biomedicine 10(1), 156–167 (2006)
[31] AlNemr, R., Meinel, C.: Getting more from reputation systems: A context-aware reputation framework based on trust centers and agent lists. In: International Multi-Conference on Computing in the Global Information Technology, pp. 137–142 (2008)

Towards Practical Enforcement Theories*

Nataliia Bielova[1], Fabio Massacci[1], and Andrea Micheletti[2]

[1] Università degli Studi di Trento, Italy
lastname@disi.unitn.it
[2] Fondazione Centro San Raffaele del Monte Tabor, e-Services for Life & Health Unit, Milano, Italy
micheletti.andrea@hsr.it

Abstract. Runtime enforcement is a common mechanism for ensuring that program executions adhere to constraints specified by a security policy. It is based on two simple ideas: the enforcement mechanism should leave good executions without changes and make sure that the bad ones got amended. From the theory side, a number of papers [6,10,12] provide the precise characterization of good executions that can be captured by a security policy and thus enforced by a specific mechanism. Unfortunately, those theories do not distinguish what happens when an execution is actually bad (the practical case). The theory only says that the outcome of enforcement mechanism should be "good" but not how far should the bad execution be changed.

If we consider a real-life example of a drug dispensation process in a hospital the notion of security automata or even edit automata would stop all requests by all doctors on all drugs and all dispensation protocols, as soon as a doctor forgot to insert the research protocol number.

In this paper we explore a set of policies called iterative properties that revises the notion of good traces in terms of repeated iterations. We start discussing how an enforcement mechanism can actually deal with bad executions (and not just only the good ones).

1 Introduction

The last few years have seen a renewed interest in the theoretical and practical aspects of the run-time security enforcement mechanisms. After Schneider's paper [11] on security automata, a number of refinements have been proposed. For example Hamlen's work on rewriting [6], and Ligatti et al. works on edit automata [2,12]. Most papers had an applied counterpart: systems actually capable of enforcing the security policies [4,7,10].

Yet, as we have shown already in [3], there is a gap between the theoretical constructions actually used in the papers and the practical implementation. The reasons behind this gap is the actual format of the definitions that have been so far used to formally describe the "good" properties of an enforcement mechanism: transparency and soundness.

* Research partly supported by the Project EU-FP7-IP-MASTER.

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 239–254, 2009.

Basically, soundness says that every output of enforcement mechanism should be valid and transparency says that in case of valid input, the output should be equal to the input. Most papers focused on kinds of good traces that be potentially enforced with particular enforcement mechanism, thus providing an initial classification. For practical applications, this is not enough. What distinguishes an enforcement mechanism is not what happens when traces are good, because nothing should happen! The interesting part is how precisely bad traces are converted into good ones. To this extent soundness only says they should be made good. The practical systems, being practical, will actually take care of correcting the bad traces. But this part is simply not reflected in the current theories. Not even Ligatti's own running example could be generated or accounted for by his theoretical constructions [3].

In order to be concrete and show the impressive width of the gap, we use a real, industrial level healthcare process set-up in some private and public hospitals for drug dispensation.

The classical enforcement mechanisms assume that as soon as some restricted operation happens, either the process must be immediately stopped [11] or the mechanism should wait until the execution becomes valid again by itself [10]. In practice a mechanism faithfully implementing security automata á la Schneider would stop the whole drug dispensation process because once a doctor did not insert a research protocol number. Following the theoretical construction actually used by Ligatti, Bauer and Walker the mechanism should wait, suppressing actions of doctors and nurses, until the doctor would insert that missing protocol number. Process becomes valid again by itself. This behavior is not inherent to the theoretical constructions that are possible with edit automata. It is just the only one that is actually provided in the cited papers. Therefore, in this paper we address the following challenge:

Challenge 1. *The enforcement mechanism should have a "plausible"/"believable" behavior when the requests (of the users) do not correspond to the policy.*

Contribution of the paper. In the next section, we describe our concrete running example: a health-care process of drug dispensation to outpatients. It will be used to show what kind of practical enforcement can be done whenever process execution violates the security policy. In §3 the basic notations (for edit automata, enforcement, etc.) that will be used in the paper are presented.

Our contributions are following: we introduce a new kind of property called iterative property. Loosely speaking, it captures the practical intuition of repeating good transactions. This property corresponds more accurately than renewal and safety property to the actual traces that are used in practice. In our practical example it covers all properties of interests with the exception of liveness.

Then, we also show the relation between iterative properties to all other classical security properties and show how to represent it with simple policy automata (§4). Finally, we show which mechanisms can concretely enforce this property and how to construct them (§6) and conclude (§7).

As a simple example, we have shown in [3] that the example of Fig.2 in [2] could not be generated from the proof of Thm.8. In this paper, this example

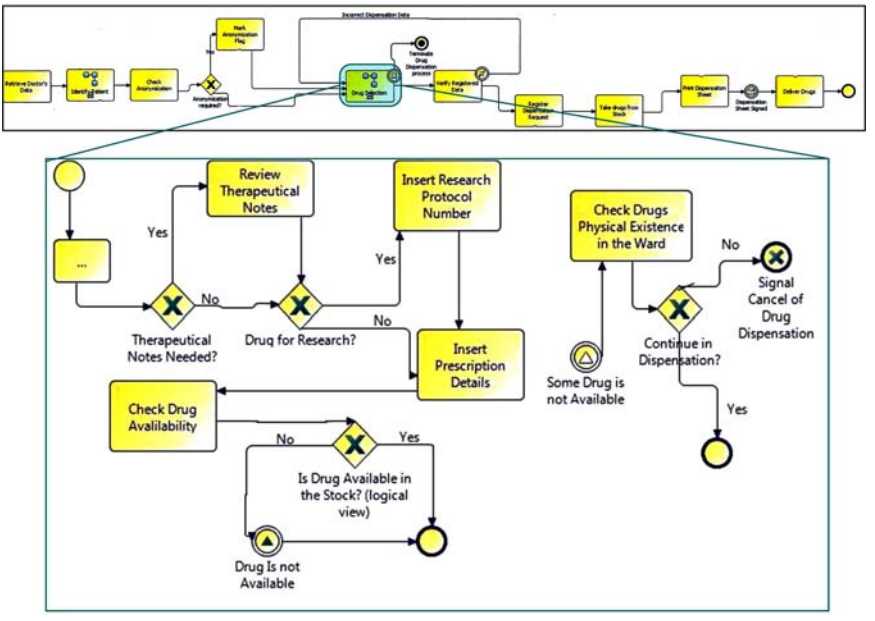


Fig. 1. BPMN model drug dispensation process

can be automatically generated from the specific automata corresponding to the policy by our algorithm in §6 if, whenever there is an attempt to violate the policy, we emit a warning instead of silently suppressing the action.

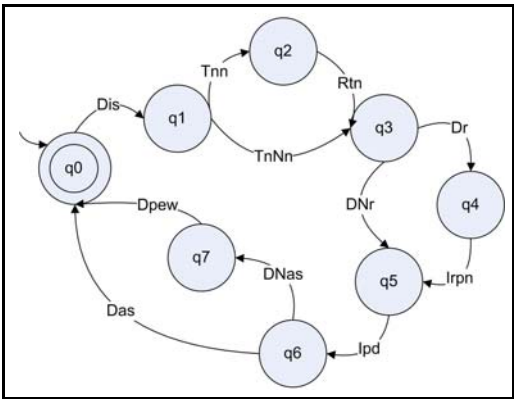
2 Running Example

The case study in this paper is based on a healthcare process of drug dispensation. Private Hospitals accredited with the Public National Health Service are charged with administering drugs or with providing diagnostic services to patients that use their structure and then are authorized to claim the cost of drug dispensation or diagnostic provisioning to the Regional Healthcare Authority.

In particular, there is a mechanism called File F that allows hospitals to refund the drugs administered and/or supplied in the hospitals' outpatient departments to the patients that are not hospitalized. Here we describe the drug dispensation process consisting of the following main steps: the doctor identifies the patient; the doctor selects the drugs, registers the dispensation, takes the drugs from the stock and prints the dispensation sheet; doctor signs the dispensation sheet and delivers drugs to the patient.

In Fig. 1 we present a BPMN diagram of drug dispensation process emphasizing the drug selection subprocess[1]. According to this subprocess, the doctor

[1] We would like to thank ANECT for developing original BPMN diagrams of the full drug dispensation process.



Abbreviations

Dis = Drug is selected
Tnn = Therapeutical notes needed
Rtn = Review therapeutical notes
TnNn = Therapeutical notes Not needed
Dr = Drug is for research
Irpn = Insert research protocol number
DNr = Drug is Not for research
Ipd = Insert prescription details
DNas = Drug is Not available in the stock
Dpew = Drug physically exists in the ward
Das = Drug is available in the stock

Fig. 2. Policy automaton for iterative drug selection subprocess

retrieves patient's prescription and dispensation historical data, selects the drugs by existing prescription, by repeating dispensation or from the stock. Then, for each selected drug the following procedure holds:

1. First the doctor should select the drug.
2. If the therapeutical notes needed, doctor reviews them in this step.
3. If patient is using prescribed drug for the research program purposes (i.e., the patient has been enrolled in the clinical trial for the testing of that drug), doctor has to insert the research protocol number into the prescription form.
4. Doctor has to insert all the other prescription details.
5. The availability of the drug in the stock is checked. If it is not available the next check is made, otherwise the process succeeds.
6. Doctor checks the drug availability in the ward. If it is available the process succeeds, otherwise fails.

In this paper we want to define how the executions where "something locally bad may happen" can be enforced. Let us come back to the Fig. 1 and assume that each choice in BPMN diagram corresponds to the action in resulting process execution that communicates the choice, e.g., if the drug is for research then the corresponding event is "Drug is for research", similarly, if drug is not for research then the event is "Drug is not for research". To ease the comparison with other papers on enforcement [2,12] we give in Fig. 2 automaton corresponding to the BPMN process. Formal definition will be given later in the section 3. Intuitively an acceptable iteration requires following the edges of the automaton and end in the state $q0$ without skipping steps.

Example 1. Let us assume the following execution of the process, which consists of 3 iterations for three different drugs. At first the drug is selected, therapeutical notes are not needed, the drug is not for research, we insert prescription details, the drug is available at the stock. The first part is correct. The second iteration is: drug is selected, therapeutical notes not needed, the drug is for research, insert prescription details, the drug is available at the stock. In this iteration the drug

is for research but the research protocol number is not inserted, therefore it is not correct. The third iteration is drug is selected, therapeutical notes needed, review therapeutical notes, the drug is not for research, insert prescription details, the drug is available at the stock. This iteration is also correct. ♢

Even if we accept the idea that incorrect execution should be dropped, the acceptable behavior for the administrators of the e-health system is just to drop the second part of the execution.

3 Security Properties

Following the standard notation on run-time security policies [11,5,2] we denote the set of observable program actions by $\mathcal{A}$. An *execution*, or a trace, is a finite or infinite sequence of actions; the set of all finite executions over $\mathcal{A}$ is denoted by $\mathcal{A}^*$, the set of infinite executions is $\mathcal{A}^\omega$, and the set of all executions is $\mathcal{A}^\infty$. We write $\tau;\sigma$ to denote concatenation of two sequences and τ must be finite. By $\tau \preceq \sigma$, or $\sigma \succeq \tau$ we denote that τ is a finite prefix of finite sequence σ.

A *security property* is a predicate $\widehat{P}$ over traces or, equivalently, a set of traces $\Sigma \subseteq \mathcal{A}^\infty$ such that $\forall \sigma \in \Sigma.\, \widehat{P}(\sigma)$. Schneider only considered infinite traces (by extending finite traces by the repetition of the last action) but we prefer to distinguish between finite and infinite traces. In the sequel the execution σ that satisfies the property $\widehat{P}$ is called *valid*, *legal* or *good*. Similarly, the execution that does not satisfy the property will be called *invalid*, *illegal* or *bad*.

There are several classes of properties. The property "nothing bad ever happens" is called *safety* property and formally defined by Lamport [8]. Additional to safety properties, there are *liveness* properties [1] that claim that "something good eventually happens during any execution" or in the other words any finite execution can always be extended to satisfy the property.

However, except for safety and liveness properties there are more general properties, which allow executions to alternate between satisfying and violating security property. *Renewal* property presented in [10] is such a property. According to it, every infinite-length sequence has infinite number of valid finite prefixes and every invalid sequence has only a finite number of valid prefixes.

Definition 1. *Property* $\widehat{P}$ *is* renewal *if the following holds:*

$$\forall \sigma \in \mathcal{A}^\omega :\ \widehat{P}(\sigma) \Longleftrightarrow (\forall \sigma' \preceq \sigma : \exists \tau \preceq \sigma : \sigma' \preceq \tau \wedge \widehat{P}(\tau)) \tag{1}$$

According to [10] every decidable renewal property can be enforced by a Ligatti automata (a particular kind of edit automata [3]). These automata will always output only the longest valid prefix of the input. The renewal property, similar to liveness property, implicitly assumes that "nothing irremediably bad happens in any finite execution". It is obviously implied by the fact that the valid execution must have infinite number of valid prefixes. Therefore if the valid execution has something irremediably bad appeared in a finite prefix, then the number of valid prefixes is finite.

Example 2. Let us expand the Ex. 1. Result of the drug selection process execution for three drugs is a following trace, which consists of three iterations:

1. The first iteration is Dis; TnNn; DNr; Ipd; Das, which is a valid execution.
2. The second attempted iteration is Dis; TnNn; Dr; Ipd; Das, which is an invalid execution. It means that the drug is for research (Dr action) but the research protocol number is not inserted (there is no action Irpn after Dr).
3. The third iteration in the trace consists of Dis; Tnn; Rtn, DNr; Ipd; Das, which is a valid iteration.

The resulting trace is invalid since it has an invalid second part after which the trace can never become valid again. What kind of behavior is expected from the enforcement mechanism in this case? ♢

The Ex. 2 clearly presents a renewal property, since if the infinite sequence is valid (always contains action Irpn after Dr or contains action DNr) it has infinite number of valid prefixes. Moreover, it can be enforced by Ligatti Automaton by outputting the longest valid prefix. In this case the resulting execution will be a first iteration of the process execution.

However, the administrators of the e-health system might expect a resulting trace to be longer. Since the drug selection process for the third drug is valid by itself, they would like to have this iteration in the resulting trace. Therefore an excected behavior of the system is following: to delete the second invalid iteration of the execution, and to output the first iteration followed by a third iteration. However, this trace correction is not provided by existing techniques [10].

The iterative drug selection process used in Ex. 2 shows that there exists a class of properties that accept set of traces consisted of repeated executions. In our example the trace is legal if it consists of iterations that are representing all the paths from state $q0$ to state $q0$ in Fig. 2. We define this class as *iterative properties* (like in [10] assuming that empty trace is always valid).

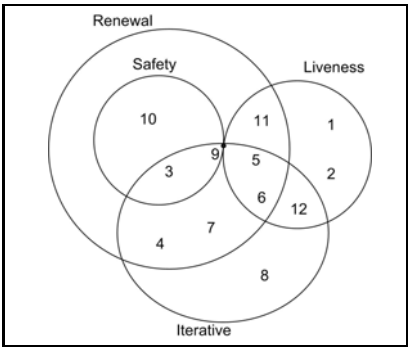
Definition 2. *Property* $\widehat{P}$ *is a* iterative property *iff*

$$\forall \sigma, \sigma' \in \mathcal{A}^*. \widehat{P}(\sigma) \wedge \widehat{P}(\sigma') \Longrightarrow \widehat{P}(\sigma; \sigma') \tag{2}$$

Notice that given an infinite sequence of the form $(\sigma)^\infty$, it can satisfy some iterative property, which accepts σ. By this example we want to emphasize, that iterative property is defined over finite and infinite sequences.

We present several examples of the iterative properties that are also renewal properties as well as some others that are not renewal and not even liveness. The relationship between safety, liveness and renewal properties is taken from Fig. 1 of [10]. We add their relationships with iterative properties in Fig. 3.

Properties 3-7 are iterative properties since by concatenating two valid sequences a valid sequence is always obtained. The trivial property 9 is iterative as well. The property 8, which was defined as non-renewal and non-liveness in [10], is an iterative property. If we concatenate two valid executions that are terminated and never access private files then the resulting execution will also be



Legend

1 Nontermination
2 Resource availability
3 Stack inspection
4 Log out and never open files
5 Property 4 on system without file-open actions
6 Eventually audits
7 Transaction property
8 Termination + file access control
9 Trivial
10 All sequences with fixed length
11 Increasingly longer sequences
12 Termination

Fig. 3. Relationships between security properties

valid. We similarly explain the property 12: since all finite sequences are valid, the resulting concatenated finite-length sequence is valid as well.

The property 10 is non-iterative. However, it is a safety property – if the sequence is invalid and its length is bigger than some fixed number n, then there exists a prefix of length $n+1$ such that any continuation of this prefix is an invalid sequence. The property 11 states that the sequence is valid iff it is infinite or its length belongs to the following set of numbers $\{F_i\}$: $F_0 = 1$, $F_{i+1} = 2F_i + 1$. This property is not iterative: by concatenating two valid sequences a new invalid sequence is obtained. Yet, this is a renewal property, because every valid infinite-length sequence has an infinite number of valid prefixes.

4 Iterative Property Representation

For finite representation of the security policies we use a variant of automata without loss of generality. Indeed, it was shown in Proposition 6.24 of [12] that edit automata can only enforce properties represented by Büchi automaton.

Definition 3 (Policy automaton). *A* Policy automaton *is a 5-tuple of the form* $\langle \mathcal{A}, Q, q_0, \delta, F \rangle$*, where* $\mathcal{A}$ *is a finite nonempty set of security-relevant program actions,* Q *is a finite set of states,* $q_0 \in Q$ *is the initial state,* $\delta : Q \times \mathcal{A} \to Q$ *is a labeled partial transition function, and* $F \subseteq Q$ *is a set of accepting states. The initial state is always an accepting state* ($q_0 \in F$) *and Policy automaton satisfies the following properties:*

1. *It does not have any dead-ends.*
2. *All the states of automaton are reachable.*
3. *It is a deterministic automaton.*

For the drug selection subprocess from Fig. 1 we constructed the Policy automaton from Fig. 2. In the following we will use this Policy automaton.

The Policy automaton can have accepting and non-accepting states. In our example, if the action Dr ("Drug is for research") happens, then the subsequent

state must be non accepting and only after action Irpn ("Insert research protocol number") the next state may be accepting.

Definition 4 (Run of a Policy automaton). *Let* $A = \langle \mathcal{A}, Q, q_0, \delta, F \rangle$ *be a policy automaton. A* run *of* A *on a finite (respectively infinite) sequence of actions* $\sigma = \langle a_0, a_1, a_2, \ldots \rangle$ *is a sequence of states* $q_{|\sigma|} = \langle q_0, q_1, q_2 \ldots \rangle$ *such that* $q_{i+1} = \delta(q_i, a_i)$. *A* finite run is accepting *if the last state of the run is an accepting state. An* infinite run is accepting *if the automaton goes through some accepting states infinitely often.*

Definition 5 (Property represented as Policy automaton). *Some property* $\widehat{P}_A$ *is represented as a Policy automaton* A *if and only if:*

$$\forall \sigma \in \mathcal{A}^{\infty} : \widehat{P}_A(\sigma) \Longleftrightarrow A \text{ accepts } \sigma \tag{3}$$

The Policy automaton combines the acceptance conditions of Büchi Automata and finite state automata. Some iterative properties can be represented as a Policy automaton. For example, a predicate $\widehat{P}$ corresponding to drug selection subprocess is an iterative property and it is represented by Policy automaton.

5 Effective vs. Iterative Enforcement

Once we have designed the security policy we can define an enforcement mechanism. Runtime enforcers transform the program executions of an untrusted application to ensure that they do not violate security property. For example, the enforcer can be formally modeled as security automata [11] (for safety properties), or as edit automata [10] (for renewal properties).

For the sake of simplicity, we only reason only about finite executions. We assume that enforcement mechanism E given some input execution sequence τ transforms it into the output sequence $E(\tau)$.

Traditionally, an enforcement mechanism should satisfy (at least) two properties: soundness and transparency. Soundness says that all the output of the enforcement mechanism should be legal. An enforcement mechanism obeys transparency if all the good sequences are output without changes. Enforcement mechanism provides "effective$_=$enforcement" [9] if it obeys both of these properties.

Definition 6. *An enforcement mechanism* E effectively$_=$enforces *a property* $\widehat{P}$ *on the system with action set* $\mathcal{A}$ *iff*

1. $\forall \sigma \in \mathcal{A}^* \,.\, \widehat{P}(E(\sigma))$
2. $\forall \sigma \in \mathcal{A}^* \,.\, \widehat{P}(\sigma) \Rightarrow E(\sigma) = \sigma$

The definition of effective$_=$enforcement is enough if we deal only with valid input. Formally speaking, a mechanism that transforms every invalid trace to an empty trace will also satisfy the property of "effective$_=$enforcement". However, none of our references from the hospital will consider an "effective enforcement"

mechanism that will stop drug selection subprocess and whole dispensation process because the doctor failed to insert a research protocol number. Therefore, sequences are not wholly good or wholly bad. They are composed by fragments that can be good or bad. The removal of a bad fragment from an otherwise good sequence can make it good.

In Thm. 4 of [3] it is shown that Ligatti automaton is specific type of edit automaton that has an all-or-nothing behavior and delayed precisely enforces given property. All-or-nothing means that it always outputs all the read actions that are not output so far or does not output anything. Delayed precise enforcement means that only the longest valid prefix is output. We give a formal definition of the longest valid prefix and type of enforcement that Ligatti automaton provides.

Definition 7. *The prefix σ_o (o for "output") is a* longest valid prefix *of the sequence of actions σ with respect to the property $\widehat{P}$ iff*

$$\widehat{P}(\sigma_o) \wedge (\forall \sigma^* . \sigma^* \succeq \sigma_o \wedge \sigma^* \preceq \sigma \Longrightarrow \neg \widehat{P}(\sigma^*)) \tag{4}$$

Definition 8. *An enforcement mechanism E* all-or-nothing delayed precisely enforces *a property $\widehat{P}$ on the system with action set $\mathcal{A}$ iff*

1. $\forall \sigma \in \mathcal{A}^* . \widehat{P}(\sigma) \Rightarrow E(\sigma) = \sigma$
2. $\forall \sigma \in \mathcal{A}^* . \neg \widehat{P}(\sigma) \Rightarrow E(\sigma) = \sigma_o$, *where σ_o is a longest valid prefix of σ.*

Let us define another type of enforcement, which is able to output longer acceptable sequences than just a longest valid prefix.

Definition 9. *An enforcement mechanism E* iteratively enforces by suppression *an iterative property $\widehat{P}$ on the system with action set $\mathcal{A}$ iff*

1. $\forall \sigma \in \mathcal{A}^* . \widehat{P}(\sigma) \Rightarrow E(\sigma) = \sigma$, *and*
2. $\forall \sigma \in \mathcal{A}^* . \neg \widehat{P}(\sigma) \wedge \exists \sigma^* \succeq \sigma . \widehat{P}(\sigma^*) \Rightarrow E(\sigma) = \sigma_o$, *where σ_o is the longest valid prefix of σ, and*
3. $\forall \sigma \in \mathcal{A}^* . \neg \widehat{P}(\sigma) \wedge \forall \sigma^* \succeq \sigma . \neg \widehat{P}(\sigma^*) \wedge \exists \sigma_b . \sigma = \sigma_o ; \sigma_b ; \sigma_r \wedge \sigma_b$ *is the smallest sequence s.t. after deleting it from σ the resulting trace can become good again $\Rightarrow E(\sigma) = \sigma_o ; E(\sigma_r)$, where σ_o is the longest valid prefix of σ.*

The example of market policy (Fig. 2 of [2]) shows exactly iterative enforcement by suppression. The only difference is that the edit automaton from Fig. 2 does not simply suppress the bad sequences but gives a warning to the user. Hence, the Ligatti automaton [3] provides all-or-nothing delayed precise enforcement, while edit automaton from Fig. 2 of [2] provides iterative enforcement by suppression. However, it is not clear how the latter automaton can be automatically constructed for given policy.

6 Iterative Enforcement Mechanism

We take the definition of edit automata from [3]. Generally speaking, edit automaton can suppress actions without further insertion, but this power of edit

automaton was not used to enforce renewal properties. It happened because the sequences corresponding to renewal property do not contain any bad parts that make them not able to become good again in the future. We present such sequence in Ex. 2, where bad part is a second iteration.

Definition 10. *An* edit automaton E *is a 5-tuple of the form* $\langle Q, q_0, \delta, \gamma_o, \gamma_k \rangle$ *with respect to some system with actions set* $\mathcal{A}$. Q *specifies possible states, and* $q_0 \in Q$ *is the initial state. The partial function* $\delta : (Q \times \mathcal{A}) \rightarrow Q$ *specifies the transition function; the partial function* $\gamma_o : (Q \times \mathcal{A}^* \times \mathcal{A}) \rightarrow \mathcal{A}^*$ *defines the output of the transition according to the current state, the sequence of actions that has been read before the current action and the current input action; the partial function* $\gamma_k : (Q \times \mathcal{A}^* \times \mathcal{A}) \rightarrow \mathcal{A}^*$ *defined the sequence that will be kept after committing the transition. The dependence between the transition, output and keep function is following: if* $\delta(q, a)$ *is defined then* $\gamma_o(q, \sigma, a)$ *and* $\gamma_k(q, \sigma, a)$ *must be defined for all* σ.

Considering σ as a sequence of actions read so far and a as input action, we write an assignment of a transition from state q to state q':

$$\boxed{\langle \delta, \gamma_o, \gamma_k \rangle (q, a) = q' \,|\, \sigma' \,|\, \sigma''}$$

where $\delta(q, a) = q'$, $\gamma_o(q, \sigma, a) = \sigma'$, $\gamma_k(q, \sigma, a) = \sigma''$.

In order for the enforcement mechanism to be effective all functions δ, γ_k and γ_o should be decidable.

Definition 11. *Let* $A = \langle Q, q_0, \delta, \gamma_o, \gamma_k \rangle$ *be an edit automaton. A* run *of automaton* A *on an* input *sequence of actions* $\sigma = \langle a_1, a_2, \ldots \rangle$ *is a sequence of pairs* $\langle (q_0, \epsilon), (q_1, \sigma_1), (q_2, \sigma_2), \ldots \rangle$ *such that* $q_{i+1} = \delta(q_i, a_{i+1})$ *and* $\sigma_{i+1} = \gamma_k(q_i, a_{i+1}, \sigma_i)$. *The* output *of automaton* A *on input* σ *is sequence of actions* $\sigma_o = \langle \sigma_1^o, \sigma_2^o, \ldots \rangle$ *such that* $\sigma_{i+1}^o = \gamma_o(q_i, a_{i+1}, \sigma_i)$.

In the sequel we use $*$ as an abbreviation for the sequence of actions kept in memory so far (actions that were read but not output yet). Then we use $*; a$ as an abbreviation for the concatenation of the current memory and action a. So $\langle \delta, \gamma_o, \gamma_k \rangle (q, a) = q' \,|\, \epsilon \,|\, *; a$ means that $\delta(q, a) = q'$, $\gamma_o(q, \sigma, a) = \epsilon$ for all $\sigma \in \mathcal{A}^*$ and $\gamma_k(q, \sigma, a) = \sigma; a$ for all $\sigma \in \mathcal{A}^*$. In order to farther simplify the reading of the figures we use boolean expressions on the edges of the automaton. So that if we label an edge with $!(a \vee b)$ it means that it can only take any action different from a or b. We use $\top$ for any action; ADD for adding an input action to the memory: $a \,|\, \epsilon \,|\, \text{ADD}$ is an abbreviation for $a \,|\, \epsilon \,|\, *; a$.

Let us come back to our Ex. 1 and to drug selection process representation in Fig. 2. First we will show how it can be all-or-nothing delayed precisely enforced and iteratively enforced by suppression. Then we compare the input and output of both types of enforcement.

6.1 Ligatti Automaton Construction

Ligatti automaton is a specific kind of edit automaton that is constructed according to the proof of the theorem about effective$_=$enforcement (Thm. 8 of [2]).

Algorithm 1. Ligatti automaton Algorithm

Input: Policy automaton $A^P = \langle \mathcal{A}, Q^P, q_0^P, \delta^P, F^P \rangle$;
Output: Ligatti automaton $E = \langle Q, q_0, \delta, \gamma_o, \gamma_k \rangle$;
1: $q_0 = q_0^P$; $Q = Q^P \cup \{q_\perp\}$;
2: **for all** $q \in Q^P \cup \{q_\perp\}$ **do**
3: **for all** actions $a \in \mathcal{A}$ **do**
4: **if** $\delta^P(q, a)$ is not defined **then**
5: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := q_\perp \mid \epsilon \mid *; a$;
6: **else**
7: **if** $\delta^P(q, a) \notin F^P$ **then**
8: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := \delta^P(q, a) \mid \epsilon \mid *; a$;
9: **else**
10: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := \delta^P(q, a) \mid *; a \mid \epsilon$;

In this section we want to show how to build such automaton, which moreover provides all-or-nothing delayed precisely enforcement.

In Alg. 1 we present our construction. As a result, we obtain an automaton which has the same behavior as an automaton constructed by the proof of Thm. 8 [2]. This proof gives a construction of infinite state automaton that keeps in the state the sequence read so far but not valid yet. Since we represent the policy as a Policy automaton, we can know whether the sequence can ever become good again. It can if transition function is defined for current state and incoming action. Otherwise the trace can never become valid again. Considering this, we show an algorithm of automaton construction that always outputs the longest valid prefix, thus providing all-or-nothing delayed precise enforcement.

Alg. 1 works as follows. Suppose the state of the automaton after executing sequence σ is q, the next incoming action is a. In line 4 we check whether there is a path in the Policy automaton from the state q on action a. If there is no such path ($\delta^P(q, a)$ is not defined) it means that there is no way to reach the accepting state of the policy, therefore the sequence can never become valid again. Hence the output done so far is the only output produced on σ and all its continuations, therefore the next state is an error state: $\delta(q, a) := q_\perp$.

Otherwise (line 6) we can reach the next state. If the next state is non-accepting, it means that possibly there is a path to the accepting state, so the currently read sequence can become good. Therefore we simply keep the current action in the memory (line 8). If the next state is accepting, then currently read input $\sigma; a$ is accepted by the Policy Automaton. Therefore, we output all the actions read so far followed by the current action (line 10). The resulting Ligatti automaton for the Policy presented in Fig. 2 is partially shown in Fig. 4 (for the sake of readability we only show outcoming edges from the states $q_4, q_6, q_\perp$).

The behavior of the Ligatti automaton constructed by this algorithm is exactly the same as of one constructed by the proof of Thm. 8. The only difference is that in Thm. 8 the state contains all the read actions and if the trace can never become good again there will be as many states as the length of the trace. While

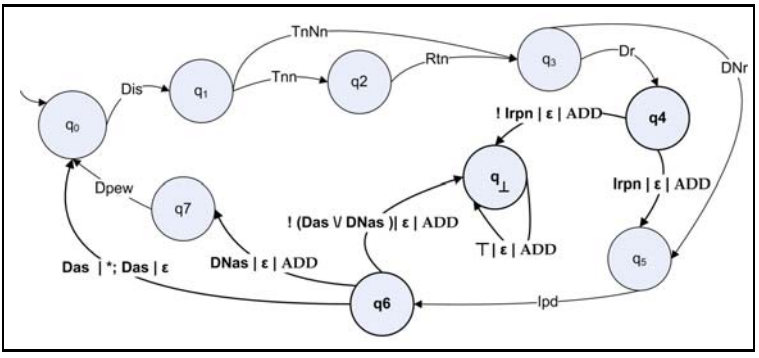


Fig. 4. Resulting Ligatti automaton for the Policy automaton from Fig. 2

in our construction, as soon as the trace cannot become good again, the next state will be an error state and all the following input actions will be kept.

Theorem 1. *A property* $\widehat{P}$ *represented by Policy automaton* A^P *can be* all-or-nothing delayed precisely enforced

6.2 Iterative Enforcement by Suppression Mechanism

We will use a following assumption for an iterative property $\widehat{P}$ represented by Policy automaton. Let's take all possible sequences of actions corresponding to all the paths form one accepting state to another accepting state such that they don't contain any accepting states in the middle. Then we assume that each of this sequences starts with a unique starting action which never appears in the sequence again. Note that several sequences can have the same unique starting action. Indeed, in the example of drug selection process (Fig. 1) every valid sequence starts as soon as doctor chooses the drug and he can choose the drug again only in the beginning of next sequence.

We can find a lot of real life examples of this kind of iterative process that start with unique starting action. For instance, remaining in the healthcare world we can consider the case when a nurse has to prepare a therapy for a hospitalized patient: once identified the specific therapy (unique starting action) the nurse starts an iterative process. Another example is the reservation of a medical examination: once the administrative personnel receives a call from a person (unique starting action), he starts a process in order to examine the request, to evaluate availability and to confirm the reservation. All these processes are repeated several times for different therapies, patients, etc.

Since every accepting sequence starts with a unique starting action, we can define the smallest bad sequence σ_b from the definition 9. Suppose, that sequence σ is executing and it corresponds to the 3rd clause of the definition. Then, as soon as the longest valid prefix σ_o executes, it is output without changes. Then, there are two possible options. First is if the next action is a unique starting action, then σ_b starts with this action; the enforcement mechanism keeps in σ_b all the

Algorithm 2. Suppressing bad parts Algorithm

Input: Policy automaton $A^P = \langle \mathcal{A}, Q^P, q_0^P, \delta^P, F^P \rangle$;
Output: Edit automaton $E = \langle Q, q_0, \delta, \gamma_o, \gamma_k \rangle$;
1: $q_0 = q_0^P$; $Q = Q^P \cup \{q_\perp\}$;
2: **for all** $q \in Q^P \cup \{q_\perp\}$ **do**
3: **for all** actions $a \in \mathcal{A}$ **do**
4: **if** $\delta^P(q, a)$ is not defined **then**
5: **if** $\exists q^A \in F^P . \delta^P(q^A, a) = q'$ **then**
6: **if** $q' \in F^P$ **then**
7: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := q' \ \ |a|\epsilon$;
8: **else**
9: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := q' \ \ |\epsilon|a$;
10: **else**
11: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := q_\perp \ \ |\epsilon|\epsilon$;
12: **else**
13: **if** $\delta^P(q, a) \notin F^P$ **then**
14: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := \delta^P(q, a) \ \ |\epsilon|*; a$;
15: **else**
16: $\langle \delta, \gamma_o, \gamma_k \rangle(q, a) := \delta^P(q, a) \ \ |*; a|\epsilon$;

actions coming after it until the new unique starting action appears in the input. At this point the new accepting sequence can start. Therefore, by deleting σ_b we make the smallest possible suppression. Second option is that the next action is not a unique starting action, then the enforcement mechanism keeps in σ_b all the actions coming after last valid input. It happens until the new unique starting action appears in the input, at this point the new accepting sequence can start. Therefore, by deleting σ_b we make the smallest possible suppression.

In this section we show how to construct an edit automaton that provides iterative enforcement by suppression of iterative property represented by Policy automaton. We propose to construct an edit automaton $E = \langle Q, q_0, \delta, \gamma_o, \gamma_k \rangle$ for iterative property $\widehat{P}$ represented by a Policy automaton $A^P = \langle \mathcal{A}, Q^P, q_0^P, \delta^P, F^P \rangle$ as shown in the Alg. 2. This algorithm is obtained from the Alg. 1 by changing the line 5 of the former algorithm to the lines 5-11 of the latter one. These changes correspond to the case when the trace cannot become good again since there are no transitions in the Policy automaton for the next incoming action.

Condition on line 5 corresponds to the case when the next incoming action a is a unique starting action. Then, if only this action is accepted by the Policy automaton, which means that $\widehat{P}(a)$ (line 6), this action is immediately output and memory is empty; next state is an accepting state corresponding to acceptance of this action. If unique starting action a is not accepted, then the memory is updated with the only this action and output empty sequence (line 9). If a is not a unique starting action (line 10), there is a transition to an error state and incoming action a is not kept in the memory. Notice, that an assumption about unique starting action is critical here, without this assumption in general case the algorithm will not satisfy the 3rd clause of the definition 9.

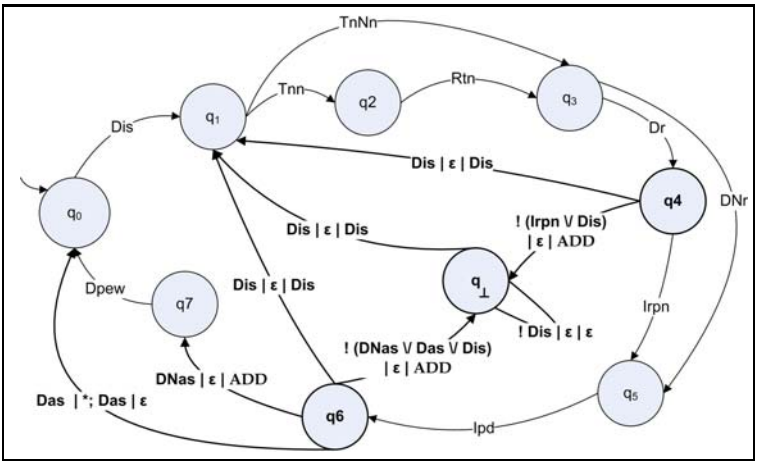


Fig. 5. Resulting edit automaton for the Policy automaton from Fig. 2

One of the differences is that the constructed automaton will be able to come back to the Policy automaton state from the error state as if no error happened. By doing so we simply delete smallest bad sequences parts (σ_b in the definition 9, which constructed automaton skips while being in the error state.

For the Policy automaton from our example an edit automaton is constructed following the Alg. 2. This automaton has 9 states but 99 transitions. For the sake of brevity we will use our abbreviations for reading the figures and show only outcoming transitions of the states $q_4, q_6, q_\perp$ in Fig. 5.

Theorem 2. *An iterative property* $\widehat{P}$ *represented by Policy automaton* A^P *can be* iteratively enforced by suppression *by the edit automaton* E *constructed by the Alg. 2.*

Let us show in Fig. 6 the output of the Ligatti automaton (that all-or-nothing delayed precisely enforces) and edit automaton (that iteratively enforces by suppression) for the same Policy automaton. The input in the figure shows 5 iterations corresponding to the drug selection process. The whole input is not valid and can never become valid again because there is a second iteration inside the trace that can never become valid again. However, third and fifth iterations are valid, which means that doctor could successfully select the drugs he wanted.

First three input iterations in the Fig. 6 are iterations from the Ex. 2. Their validity can be checked by the Policy automaton shown in Fig. 2. The Ligatti automaton shown in Fig. 4 outputs the longest valid prefix, hence only the first iteration. It means that doctor will successfully complete selection process only for the first drug. Edit automaton shown in Fig. 5 will output all the successful iterations, which means it outputs first, third and fifth iterations.

If in Alg. 2 at lines 7, 9 and 11 we prefix the current output with the "warning" string we would get the edit automaton from Fig. 2 in [2]. This time it's made not manually but automatically.

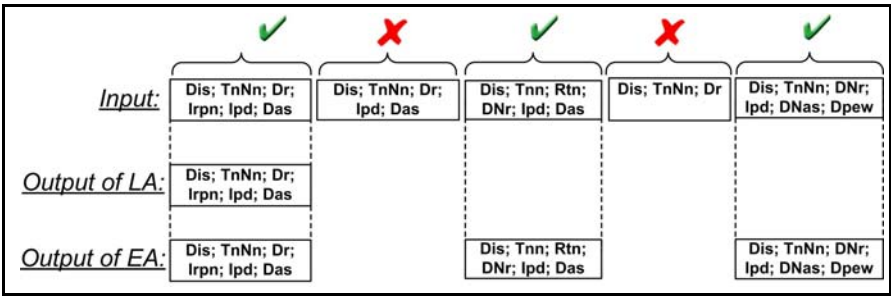


Fig. 6. Output of Ligatti (LA) and edit (EA) automata

7 Conclusions

Runtime enforcement is based on two simple ideas: the enforcement mechanism should leave good traces without changes and make sure that the bad ones got amended. From the theory side, a number of papers [6,10,12] provide the precise characterization of good executions that can be captured by a security policy and thus enforced by a specific mechanism. Unfortunately, those theories do not distinguish what happens when an execution is actually bad (the practical case). The theory only says that the outcome of enforcement mechanism should be "good" but not how far should the bad execution be changed.

If we consider a real-life example of a drug dispensation process in a hospital the notion of security automata or even edit automata would stop all requests by all doctors on all drugs and all dispensation protocols, as soon as a doctor forgot to insert the research protocol number.

In this paper we have shown a notion of enforcement that makes it possible to overcome the distinction between bad traces and good traces. The theoretical research on enforcement [6,10,12] only offered two properties: the mechanism should leave good traces untouched and make sure bad traces got amended. How they are amended was never formally specified. Notice that this was in contrast to the actually implemented systems. For instance Polymer did allow traces to be amended. But this was not reflected in the formal theory.

By revising the notion of good traces in terms of iterations we can offer a formal characterization of how enforcement mechanism deals with the bad traces.

These are the first steps towards closing the gap between the current theoretical works and their practical implementations. As a modest, but still telling example, the running example of Fig. 2 in [2] could not be generated from the policy by any of the formal construction appeared in that paper, nor in the constructions appearing in later papers. In contrast, it can be obtained by our second algorithm if, instead of suppressing every bad iteration, we emit a warning.

As a future work we consider the following. There are some actions that cannot be fixed in real life. We call these actions *observable* actions. This aspect is very important in real life when it involves the interactions of an organization with another one; for instance we can refer to the cases of outsourcing services or to

the cases in which some requests are done to external parties. Moreover it is important to know that it is possible to have observable actions also within an organization; for example there are some physical actions that cannot be deleted. For instance when a doctor or a nurse, preparing a set of therapeutical drugs for a specific patient, takes a wrong drug from a stock it is not possible to delete this physical action with a given theoretical approach.

In the future a model for observable and fixable actions can be built, then it should be decided how the enforcement can be done under this assumption. For example, edit automaton can provide enforcement by insertion but cannot suppress observable actions.

We will also consider the case of multiple users and behavior of enforcement mechanism in that case. Indeed, many doctors may try to dispense drugs at the same time, and construction of enforcement mechanism can be different. We leave this problem for the future work.

References

1. Alpern, B., Schneider, F.B.: Defining liveness. Inform. Processing Letters 21(4), 181–185 (1985)
2. Bauer, L., Ligatti, J., Walker, D.: Edit automata: Enforcement mechanisms for run-time security policies. Int. J. of Inform. Sec. 4(1-2), 2–16 (2005)
3. Bielova, N., Massacci, F.: Do you really mean what you actually enforced? In: Proc. of the FAST 2009 workshop, vol. 5491, pp. 287–301. Springer, Heidelberg (2008)
4. Erlingsson, U.: The Inlined Reference Monitor Approach to Security Policy Enforcement. Technical report 2003-1916, Department of Computer Science, Cornell University (2003)
5. Fong, P.W.L.: Access control by tracking shallow execution history. In: Proc. of Symp. on Sec. and Privacy, pp. 43–55. IEEE, Los Alamitos (2004)
6. Hamlen, K.W., Morrisett, G., Schneider, F.B.: Computability classes for enforcement mechanisms. TOPLAS 28(1), 175–205 (2006)
7. Hamlen, K.W., Morrisett, G., Schneider, F.B.: Certified in-lined reference monitoring on .net. In: Proc. of the workshop on Prog. Lang. and analysis for security (PLAS 2006), pp. 7–16. ACM Press, New York (2006)
8. Lamport, L.: Proving the correctness of multiprocess programs. TSE SE-3(2), 125–143 (1977)
9. Ligatti, J.: Policy Enforcement via Program Monitoring. PhD thesis, Princeton University (2006)
10. Ligatti, J., Bauer, L., Walker, D.: Run-time enforcement of nonsafety policies. ACM Transactions on Inf. and Sys. Sec. (TISSEC) 12(3), 1–41 (2009)
11. Schneider, F.B.: Enforceable security policies. ACM Transactions on Inf. and Sys. Sec. (TISSEC) 3(1), 30–50 (2000)
12. Talhi, C., Tawbi, N., Debbabi, M.: Execution monitoring enforcement under memory-limitation constraints. Inform. and Comp. 206(2-4), 158–184 (2007)

Security Analysis of AN.ON's Payment Scheme

Benedikt Westermann

Center for Quantifiable Quality of Service*
Norwegian University of Science and Technology

Abstract. In recent years several payment schemes have emerged for anonymous communication systems such as AN.ON and Tor.

In this paper we briefly present a payment scheme that is deployed and currently used by AN.ON. The main contribution of this paper is a security analysis of the most important cryptographic protocols involved in the payment process. The analysis of the protocols shows that they contain several weaknesses that need to be addressed to provide a fair service. We show how an attacker can use the weaknesses to surf on other's credits. Finally, we propose a fix for the protocols in order to withstand the encountered attacks.

1 Introduction

Many publications in the area of anonymous communication have been published in the last decade. Most of them deal with anonymous routing mechanisms or with attacks on the latter. Besides theoretical work they have also deployed anonymization networks based on the theoretical work. Two of the most widespread deployed anonymization systems are Tor[7] and AN.ON[2].

For both systems it is understandably crucial to find operators willing to cover the costs and the possible legal consequences of providing an anonymization service. The Tor project is not commercial, therefore it can not pay the operators itself. Thus, Tor strongly depends on volunteers willing to provide servers in the network. Clearly, these circumstances make it more difficult to find operators for the Tor network. To overcome these difficulties a payment protocol for anonymous routing was proposed by Elli Androulaki et. al.[1].

In the case of AN.ON, a spin-off company named JonDos[1] was founded in 2007. Its business model is to act as broker between the operators and the users of the network. The company provides the operators of mixes, the service providing network nodes, with the possibility of collecting money from the users for the offered anonymization. This possibility creates an incentive for people to become an operator for AN.ON.

Clearly, payment protocols must fulfill various requirements. One requirement is that the payment process does not compromise the anonymity of users. Another requirement is that the payment process must be fair, so none of the parties

* "Center for Quantifiable Quality of Service in Communication Systems, Center of Excellence" appointed by The Research Council of Norway, funded by the Research Council, NTNU and UNINETT. http://www.q2s.ntnu.no

[1] https://www.jondos.de

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 255–270, 2009.

should be able to betray other participants. To the best of our knowledge there is no security analysis for AN.ON's payment system available. Thus, we will take a closer look on the used cryptographic protocols.

In this paper we formalize and examine the cryptographic protocols which have been used since 2007 for the payment procedure in AN.ON. Our main focus is to check whether the protocols fulfill their basic tasks or fail to do so. In the latter case we also present some consequences of the encountered weaknesses.

This paper is divided into six sections. Section 2 describes the anonymization concept of AN.ON and the idea of its payment concept. In Section 3 we present our assumptions and our notation. Section 4 deals with the cryptographic protocols involved in the payment process, as well as some attacks on the protocols. The reasons for the attacks and possible countermeasures are discussed in Section 5. Section 6 concludes the paper.

2 Description of AN.ON

2.1 Anonymization Process

An objective of AN.ON's anonymization system is to provide *sender anonymity* [9] on the network layer for its users versus the receivers of messages. This means that a receiver of a message cannot determine the network address, for example the IP address, of a sender. In addition, AN.ON aims to provide *relationship anonymity* with respect to the operators of AN.ON. Therefore, no operator can determine which entities are communicating with each other.

In order to fulfill these objectives, so-called *mix* servers, also called *mixes*, are introduced. The general task of a mix is to remove correlations between incoming and outgoing messages [6]. In the deployed AN.ON system this is achieved solely[2] by multiple layers of encryption (Fig. 1). A single layer of encryption is added and removed respectively by a single mix.

A single mix between a sender and a receiver is not sufficient in order to protect the users against the operator of a mix, since an operator is able to link incoming and outgoing messages. Thereby he can uncover the relationship between sender and receiver. This is avoided by chaining several mixes together (see Fig. 1). Such a chain of mixes is called a *cascade*. Only if all operators in a cascade collude they are able to revoke the user's relationship anonymity.

It is important to notice that the order of the mixes in a chain is fix and is chosen by the operators of the mixes. Therefore, a user cannot freely choose the order of the mixes in a cascade. This has various advantages as well as disadvantages [3]. One advantage is that all mixes in the cascade process the same amount of packets. Another advantage is that each packet is sent along the same route through the cascade. Both properties are mandatory for AN.ON's payment scheme.

[2] Even though mixing of packets was mentioned in the original publication, it is nowadays deactivated due to performance reasons.

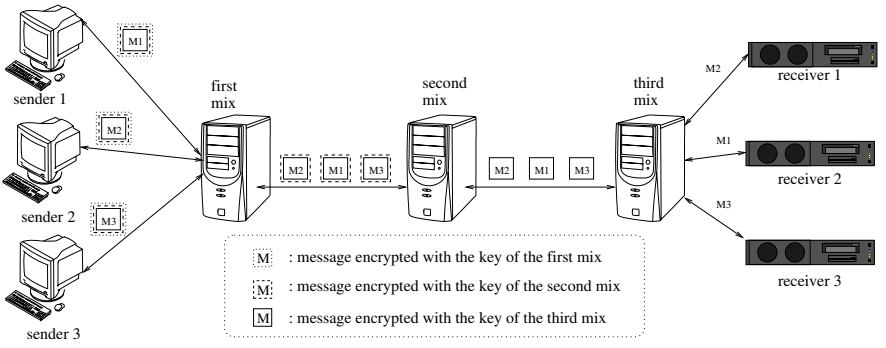


Fig. 1. Example for a cascade in AN.ON

2.2 Payment Concept

Cascades can be divided into free cascades and premium cascades. Free cascades can be used by every user without any fee. However, they are often only operated by one single operator. The free cascades do not guarantee any quality of service. By contrast, premium cascades promise a minimum bandwidth for their users. Beyond that premium cascades are more secure due to distributed and dedicated servers as well as certified and independent operators.

Certainly, the additional service of the premium cascades is not for free. To collect money from the users AN.ON has integrated a payment system which bases upon a prepaid concept [10,8]. It is similar to the concept which is used in a phone booth. When users connect to the cascade, they pay a small amount of money in advance (up to 3 EuroCent) to the operators in order to transfer some bytes (3 MB) over the cascade. Once they have almost consumed the bought traffic volume they are able to purchase new traffic volume on the fly at the cascade. The purchase of new traffic volume is possible without the termination of existing connections.

However, the transfer of such small amounts of money is not efficient because of the overhead and the lack of digital money. In order to solve the problem, users buy some credit[3] at a so-called *payment instance*. The payment instance is operated by a trusted third party. Afterwards, users can spend their bought credits on a premium cascade.

The first mix in a premium cascade is responsible for the accounting. Thus, we call the first mix also accounting instance[4]. Due to the fact that every mix in a cascade transfers the same amount of data, it is possible to implement the accounting only at the first mix. Hence, behind the first mix the packets which are related to the traffic of users are indistinguishable from packets in a free cascade.

[3] Current rates are between 2 Euro for 200 MB up to 40 Euro for 6,5 GB.
[4] Accounting instances and payment instances are operated by different operators.

3 Assumptions and Notations

We assume that the payment instance is trustworthy with respect to the payment. However, it is not trustworthy with respect to the anonymization process. Furthermore, we assume that every participant in the protocol knows the public key of the payment instance as well as the public key of the accounting instance. Concerning the accounting instance, we assume that it may act malicious in the payment process.

Cryptographic keys in this paper are denoted with prefix K. K_x represents the public key of a principal X. K_x^{-1} is the corresponding private key of the principal X. In particular we assume that if a principal X owns K_x^{-1} he automatically possesses K_x. K_{xy} denotes a session key between a principals X and Y. The term $X \ni a, b, c$ denotes that the principal X is in possession of the items a, b, c.

We assume a non-global but active adversary who cannot break cryptographic primitives. Our assumed attacker model is equal to the attacker AN.ON can withstand in its current implementation.

4 A Closer Look on the Protocols

4.1 Overview over the Protocols

AN.ON involves various cryptographic protocols in order to establish the payment process. This section describes in detail the most important protocols with their assumptions, intentions and weaknesses.

Fig. 2 gives an overview over some protocols and parties which are involved in the payment process. The protocols between the payment instance and the accounting instance are integrated within the *payment initialization protocol* and the *traffic repurchase protocol* respectively. Thus, both protocols are not described separately. Please note, there are also protocols concerning the payment process between mixes, but those remain unaccounted for. The following text provides a detailed description of the protocols mentioned in Fig. 2.

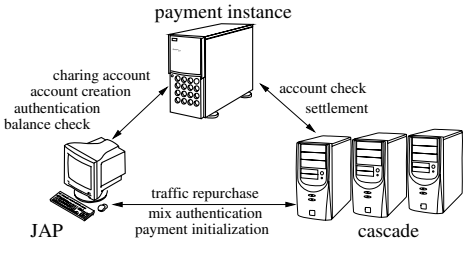


Fig. 2. Account creation protocol

4.2 Account Creation

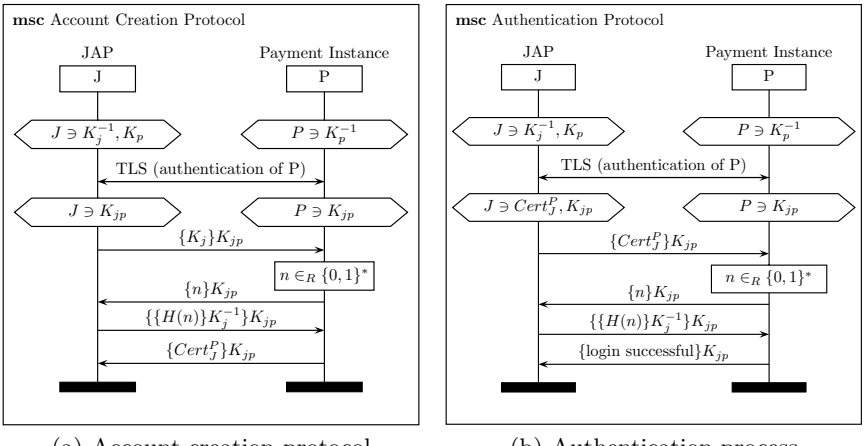
The *account creation protocol* involves a payment instance and a user which is in AN.ON normally called *JAP*. The objective of this protocol is to create an account for the user. An account is bound via a private key to the user. The private key is created by the user prior to the protocol run. Only with the private key a user is later on able to prove her ownership of the account and subsequently buy traffic volume from an accounting instance.

In order to attest that a public key belongs to a valid, but maybe uncharged account, the payment instances issue a certificate for the public key. The certificate should only be issued by the payment instance if it is convinced that a corresponding private key exists.

Fig. 3(a) presents a message sequence chart of the account creation protocol. We assume the following preconditions for a protocol run. Firstly, the payment instance possesses its own key pair (K_p, K_p^{-1}). Secondly, a user holds the public key (K_p) of the payment instance. Lastly, she also owns a key pair which has been freshly generated by herself.

The protocol starts with an establishment of a TLS connection. We assume that an establishment of a TLS connection results in a fresh symmetric key which is known to both parties. It is important to mention that only the user authenticates the payment instance.

After the establishment process of the TLS connection the user sends her public key encrypted with the session key to the payment instance. The payment instance checks if the key has already been used. If the public key is unknown to the payment instance, it generates a random challenge which is encapsulated in a XML message. After its encryption the encrypted XML messages is sent back to the user. The user hashes the decrypted message including the `<DontPanic>` element of the challenge (see Fig. 4). In the following step the user signs the



(a) Account creation protocol (b) Authentication process

Fig. 3. Two important protocols between a user and a payment instance

```
<Challenge><DontPanic version="1.0">iNYpjACH7[..]</DontPanic></Challenge>
```

Fig. 4. An example for an unencrypted challenge used in the protocols

result with her private key and sends it back to the payment instance. The response to the challenge aims to convince the payment instance that a private key exists which corresponds to the transmitted public key. Thus, if the payment instance is able to verify the signature of the hash, it can assume that a private key exists which corresponds to the received public key. However, this does not necessarily mean that the corresponding communication party owns that key. If the signature is verified successfully the payment instance creates an account and a corresponding account certificate (Fig. 5) which is signed by the payment instance with its private key K_p^{-1}. The account certificate includes the ID of the payment instance, an ID representing the account, a timestamp and the public key K_j. The certificate is sent to the user encrypted with K_{jp}. We denote the account certificate with $Cert_J^P$. The subscript indicates the owner of the certificate/private key and the superscript shows who has signed the certificate.

```
<AccountCertificate version="1.0">
    <AccountNumber>160520966610</AccountNumber>
    <BiID>ECD365B98453B316B210B55546731F52DA445F40</BiID>
    <CreationTime>2009-04-01 13:37:29.968</CreationTime>
    <JapPublicKey version="1.0">[..]</JapPublicKey>
    <Signature>[..]</Signature>
</AccountCertificate>
```

Fig. 5. An example of an account certificate

A closer look on the protocol shows that the user never proves the possession of the session key. This leads to the problem that the session key and the user's public are independent. Thus, a payment instance cannot know if the person who possesses the session key also owns the key pair (K_j, K_j^{-1}). The general problem is depicted in Fig. 6 for the authentication protocol which is discussed in the following part.

4.3 Authentication at the Payment Instance

The first *authentication* protocol takes place between a payment instance and a user. Its objective[5] is to do a mutual authentication between both parties. As precondition we assume that a user possesses an account certificate. Additionally, we assume the payment instance to be responsible for the user's account. Fig. 3(b) shows the protocol with the help of a message sequence chart.

[5] The objectives of the protocols are not stated in the literature, therefore we need to assume reasonable objectives for the protocols.

The protocol starts, similar to the previous protocol with the establishment of a TLS connection. During the establishment phase only[6] the user authenticates her communication partner. The execution of a TLS handshake results in a fresh session key known to both parties.

In the next step the user sends the signed account certificate to the payment instance which verifies the certificate. If this was successful the payment instance creates a challenge and sends it encrypted with K_{jp} to the user. When the user receives the message she decrypts the challenge, creates a hash of the message and signs the result with her private key. Afterwards the user sends the result encrypted with K_{jp} to the payment instance. After the payment instance has checked the result, an encrypted confirmation is sent back to the user. This facilitates the authentication process between the payment instance and the JAP.

An analysis of the protocol shows that it faces the same problem as the *account creation protocol* (see Fig. 4.2). It does not guarantee that the session key and the private key are owned by the same principal. Due to this a third party can claim the ownership of a user's account even if she does not own the private key (see Sec. 4.6). Fig. 6 points out the general problem of the protocol.

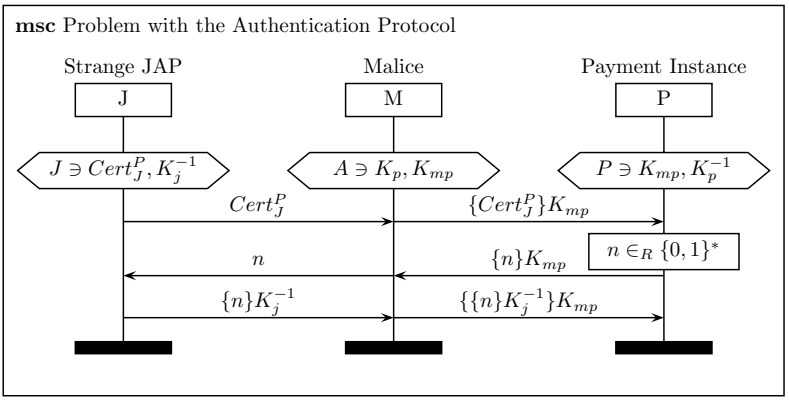


Fig. 6. The general problem of the authentication protocol

4.4 Charging of the Account and Balance Check

The *charging account* as well as the *balance check* protocol are built on top of the authentication protocol. Thus, both protocols assume that each party owns the session key. Additionally, they require that both participants are mutually authenticated. All messages which are exchanged during the protocol runs are encrypted with the afore negotiated session key.

[6] Please note, we are not the authors of this protocol, we only describe the protocol. Thus, the reasons for the design are unknown to us.

The purpose of the *charging account* protocol is to provide the user with a transaction ID. The transaction ID can be used by the user as subject for the necessary money transfer, e.g. by a bank transfer. In the rest of the paper this protocol remains unaccounted for.

In the case of the *balance check* the user sends a *balance check request* to the payment instance. As a result she obtains a document stating the current status of the account. The document includes amongst others the amount of used traffic, the remaining credits and a signature of the payment instance confirming the correctness of the data.

4.5 Mix Authentication

The protocol takes place between users and an accounting instance. Since this paper deals with the payment protocols of AN.ON, an analysis of the underlying anonymization and authentication protocols is out of scope. We assume that they work correctly and that the authentication protocol authenticates the mix. As for the TLS protocol we assume that it results in a shared and valid session key between both parties. In addition we assume that only the mix side is authenticated. If the assumptions for the mix authentication protocol are correct is subject of further research.

For the mix authentication only the public keys of the payment instance and the accounting instance are required on the user's side. It does not require any secret on the side of the user.

4.6 Payment Initialization

Presentation of the Payment Initialization Protocol. The *payment initialization* protocol requires a successful run of the *mix authentication* protocol.

A run of this protocol aims to verify that a user is in the possession of a valid and charged user account. Hence, a user needs to provide an accounting instance with the user's *account certificate*. In addition, a user needs to prove the possession of the corresponding private key. In order to verify that a user's account is charged the protocol requires also the involvement of a payment instance. This is due to the fact that an accounting instance does not have the knowledge about the current status of a user's account.

Fig. 7 presents the protocol between an accounting instance and a user. The protocol requires a user to possess an own key pair, an account certificate for the public key, the public key of the payment and the accounting instance. A payment instance only knows the public key of the accounting instance, the user's public key and its own key pair. Lastly, an accounting instance needs to own its key pair and the public key of the payment instance. In Fig. 7 it also holds a so-called *cost confirmation*. Such a cost confirmation[7] attests that the user has bought traffic from the accounting instance. It contains the overall traffic which a user has consumed on a cascade, an ID of the user's account, an ID of the

[7] It can also be seen as a digital coin.

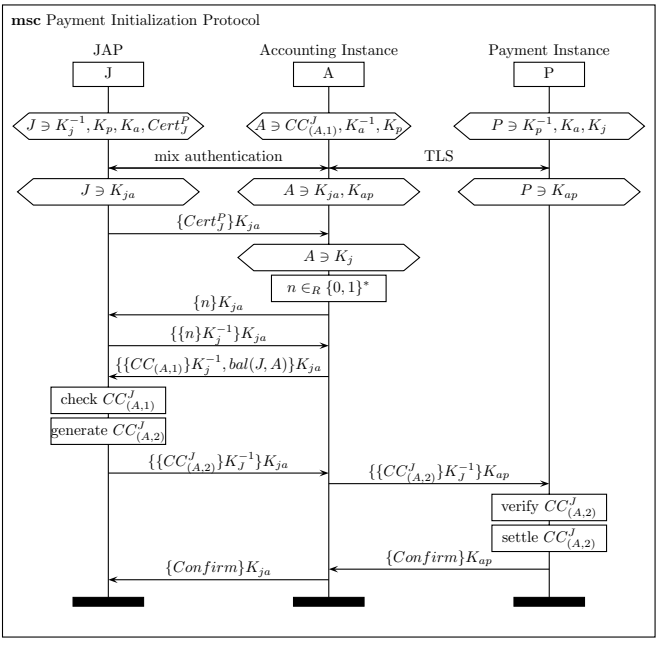


Fig. 7. The payment initialization protocol

responsible payment instance and an ID of the cascade for which the certificate was issued. Additionally, it contains a list of hashes referring to *price certificates* of the mixes in a cascade. Finally, it includes a signature of the user.

Price certificates or the hashes of them are needed for the accounting process between a payment instance and the operators of a cascade. It indicates how much money each operator of the cascade receives by anonymizing a GB of traffic. However, the price a user pays for each anonymized GB does not depend on the certificates, it solely depends on the rate which was agreed between the user and the payment instance. Thus, we do not discuss this in greater detail.

For the protocol run in Fig. 7 we assume that the user already bought some traffic on the cascade in a prior session. Thus, the accounting instance already holds an old cost confirmation of the user. A cost confirmation is denoted by $CC_{(X,n)}^Y$. It means that the cost confirmation is issued by a principal Y for a principal X and n is a sequence number for the confirmations. The higher the sequence number the more current is the confirmation.

In the first part of the protocol two session keys are created between the participants. The first session key (K_{ja}) is established between the JAP and the accounting instance with the help of the *mix authentication protocol*. The second one (K_{ap}) is created with a TLS handshake between the accounting instance and the payment instance.

Please note, only the accounting instance verifies its communication partner during the second key establishment process. Thus, the payment instance does

not know if it talks with the accounting instance. For both keys (K_{ap}, K_{ja}) we assume that they are valid and uncompromised.

In the following steps the user sends her account certificate encrypted with the session key to the accounting instance. The accounting instance checks the signature of the certificate with the public key of the payment instance. If this succeeds the accounting instance generates a random challenge and sends it to the user. After the user has received the challenge she signs the challenge with her private key (K_j^{-1}). Subsequently, she sends the result back to the accounting instance which verifies the user's signature. If this also succeeds the accounting instance needs to distinguish two cases.

In the first case the user has already used the cascade. Thus, the user has already issued a cost confirmation which can be fetched from the database of the accounting instance. Additionally, the accounting instance retrieves the amount of unused traffic $(bal(J, A))$ which has been bought in a previous session. Both information will be sent encapsulated in a so-called *pay request* to the user. When the user receives the pay request she compares the provided information with her own information. If both information correspond to each other the user creates a new signed cost confirmation on basis of the information.

In the second case the user uses the cascade for the first time. Due to this the accounting does not hold any former cost confirmation which it can present to the user. Instead of an old signed cost confirmation the accounting instance sends a *unsigned cost confirmation*. The unsigned cost confirmation informs the user about some information which needs to be included in the signed version. Basically, this includes the price certificates of the mixes within the cascade, the account number, the ID of the cascade and the number of bytes the user has to buy in advance. The latter needs to be verified by the user if it is a valid (and acceptable) value. If this value exceeds an upper bound[8] the user will not continue the payment initialization process. In addition to the proposed values the user needs to add the ID of the responsible payment instance to the unsigned cost confirmation. Now she only needs to sign the prepared document. Once this is also done, the user has created the initial cost confirmation.

After everything is checked, created and signed the user sends the cost confirmation back to the accounting instance. The accounting instance verifies the signature of the confirmation and checks if the provided information matching the minimum requirements with respect to the amount of prepaid bytes.

Due to the fact that the accounting instance is not able to check the balance of the user's account, it needs to contact the payment instance. The payment instance can decide with the help of a cost confirmation if the account has enough credits left. In order to consult the payment instance, the accounting instance forwards the cost confirmation to the payment instance.

The payment instance verifies the cost confirmation and checks if the account of the user has enough credits left to settle the received cost confirmation. If this is the case the payment instance sends back a confirmation that the cost confirmation was valid and is now settled. When the accounting instance has

[8] Currently this upper bound is at 3 MB.

received a confirmation that the cost confirmation was valid, it confirms this to the user by sending a confirmation message: `<LoginConfirmation code="0">AI login successful</LoginConfirmation>`. This confirmation is also encrypted with the session key.

Analysis of the payment initialization protocol. During the payment initialization phase the accounting instance provides the user with the unused but paid traffic volume. This amount of bytes is represented by $bal(J, A)$ in Fig. 7. A possible problem arises if the user does not track the number of bytes which are paid but haven't been used. In this case the accounting instance is able to betray the user. However, we will not address this possible problem in this paper.

Another issue arises due to the fact that in the current protocol version the confirmation of the payment instance is not linked to the cost confirmation which was handed in by the accounting instance. Thus, a replay might be possible under some special conditions, e.g.:

1. the accounting instance hands in multiple cost confirmations during a single TLS session, and
2. the block cipher operates in the *electronic code book(ECB)* mode.

Under these conditions an attacker is able to replay an old confirmation of the payment instance to betray the accounting instance. We believe that it is risky to rely on such details. A possibility to address this is to return a document which refers uniquely to the cost confirmation. The document is signed by the payment instance. This improves the protocol in two ways. Firstly, it circumvents a replay attack, sine the message cannot be used to acknowledge another cost confirmation. Secondly, a signed document gives the operator the possibility to proof later on that the payment instance has confirmed the cost confirmation. Another advantage might be that the encryption between the payment instance and the accounting instance becomes unnecessary. However, this needs to be checked in more detail.

A closer look on this protocol shows that it uses the same authentication process as the authentication protocol between the user and the payment instance. In combination with the problem sketched in Fig. 6 the accounting instance is able to claim the identity of a user.

In order to mount the attack, the accounting instance needs to wait until a user connects to the cascade. Fig. 8 sketches the attack. As in the normal protocol, the process starts with the mix authentication process. The accounting instance authenticates itself to the user. Up to here the accounting instance behaves like an honest one. The attack starts after the user has provided the accounting instance with the user's account certificate. Instead of generating a random challenge, the accounting instance establishes a TLS connection to the payment instance which is referred to in the account certificate. Note, for this operation the malicious accounting instance does not need any secret data.

When the TLS connection is established the accounting instance simply forwards the received account certificate of the user to the payment instance. The payment instance answers in conformance with the protocol with a challenge.

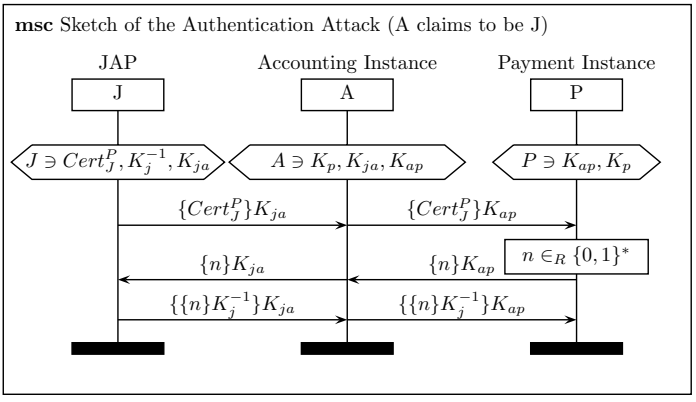


Fig. 8. Accounting instances claims to be the user

The challenge is forwarded by the accounting instance to the user who subsequently provides the accounting instance with a signed version of the challenge. The answer is forwarded by the accounting instance to the payment instance. Since the response of the challenge includes a valid signature of the user, the payment instance will confirm the successful login to the accounting instance. Thus, the accounting instance is authenticated in the name of the user even though it is not in possession of the user's private key.

Due to the authentication at the payment instance the attacker is now able to retrieve the user's account status. Such an account status contains the number of bytes the user has consumed on the different cascades. In addition it includes the last cost confirmations the user has issued for the different cascades. If any of the cost confirmation has been created without the consumption of traffic[9], the accounting instance is able to use the cost confirmation to consume the bought traffic of the user. In order to do this the malicious accounting instance needs to wait until the user connects again to the cascade. Afterwards it can mount a similar attack as the one described by Fig. 8. The only difference is that the accounting instance does not connect to the payment instance, but rather to the accounting instance which is referred to in the cost confirmation. In section 5 we will deal with the underlying reasons for the attacks.

4.7 Data Exchange and Repurchase of Traffic Volume

The protocol is built upon the payment initialization protocol. Thus, it assumes a shared session key and some amount of confirmed prepaid volume for a user. Therefore the user is able to transfer mix packets over the cascade. The client as well as the accounting instance keep track of the number of packets which are transmitted over the cascade.

[9] This is the case, if the user connects to the cascade but does not send any further packet.

When the amount of prepaid bytes drops under a predefined *lower limit* the accounting instance sends a new pay request. The pay request is similar to the pay request which is sent during the very first connection to the cascade. Thus, the pay request contains the account number, the ID of the payment instance, the price certificates of the cascade and the (minimum) number of bytes a user needs to confirm. The pay request notifies the client to send a new cost confirmation in order to further use the cascade.

As soon as the client receives such a pay request it checks the received data, adds the ID of the responsible payment instance and compares the number of bytes which need to be confirmed. Normally, the user has already transmitted additional packets since the pay request was issued. Therefore the user usually confirms more than the pay request requires. The result of this process is signed with the user's private key and sent back to the accounting instance.

In contrast to the payment instance the accounting instance can choose whether it forwards the new cost confirmation directly to the payment instance or it waits to submit a later cost confirmation to the payment instance. In the latter case the accounting instance only needs to hand-in the latest cost confirmation. Obviously, this is a trade-off between resources which are needed and the possible loss which arises if a client is malicious. However, the safest way is to submit the cost confirmation directly. Hence, the protocol is equal to the last part of the protocol in Fig. 7. Although, there are some interesting problems, for example what happens if the confirmation of the payment instance arrives after the client's volume is consumed. We will not further discuss them, since they do not directly affect the protocol.

5 Reasons for the Attack

In the previous section we pointed out some weaknesses of the protocols. The first weakness of all the authentication protocols is that a payment instance and an accounting instance respectively does not know if the session key is possessed by the same person who also owns the private key to the account. The second problem is the fact that the challenge-response procedure is transferable to other protocols. The combination of both weaknesses enables an accounting instance to claim the user's identity and at a later stage consume some of the bought traffic. However, the attack is not only restricted to the accounting instance. Also a malicious payment instance is able to mount the attack. In order to fix the protocols both weaknesses must be addressed.

The transferability of the challenge-response procedure can be avoided by the modification of the response. We propose to add the ID of the challenger and an identifier of the protocol which uniquely identifies the protocol and its version. We justify this with the following reasons. Firstly, an ID of the protocol avoids that the challenge can be transfered to another protocol or to an older versions of the protocol. Secondly, the ID of the challenger is added to the response to avoid that another malicious instance can use the response in another run of the protocol, for example the malicious instance can choose the same key as the JAP and therefore replay the messages. This attack would be similar to the one in

Fig. 8. However, we do not add the responder's ID since this would not strengthen the properties of the protocol. This is due to two reasons. Firstly, an attacker who is able to read the challenge is probably also able to arbitrarily change the unprotected challenge. Hence, he can also substitute the ID of the responder by the false one. Secondly, the signature in the response implicitly decodes the ID of the responder. If the verifier of the response is honest then he would reject the response to the challenge if it is not verifiable with responder's public key. If the verifier is malicious he can accept the response anyway. We added the information to the response and not to the challenge, since an additional information in the challenge brings no additional security. Nevertheless, it might be useful to add the information also to the challenge due to practical reasons.

In the following we address the problem that arises due to the independence of the session key and the private key. One solution is to use the client authentication capabilities of TLS. Unfortunately, this is not a solution for every used protocol of the payment concept. For instance the payment initialization protocol (and the underlying mix authentication protocol) is not based on TLS.

Another solution is to modify the response in the challenge-response procedure in order to bound the session key to the user's public key. Instead that the user hashes only the "normal" response, she can additionally include the session key to the information which she needs to hash and sign. The modified version with all of our changes is shown in Fig. 9.

In Fig. 9 the *initialization protocol* represents the messages which are necessary to authenticate the server and to transmit the user's public key. The server is either the payment instance or the accounting instance. The fixed version of the challenge-response procedure should replace the challenge-response parts of

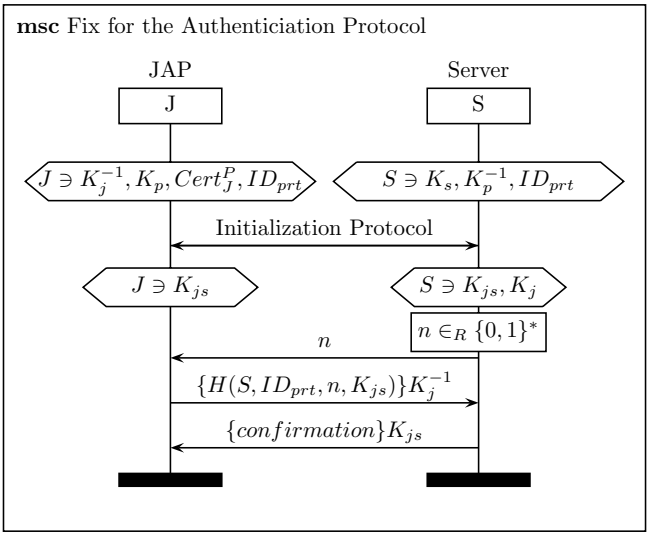


Fig. 9. A fix for the (user) authentication protocols

the protocols presented in previous section. It mainly fixes the two encountered weaknesses of the authentication protocols. In addition we have skipped unnecessary cryptographic operations. Neither the challenge nor the response needs to be encrypted with the session key K_{js}. The *confirmation* message represents the different possible replies of the server. Obviously, it depends on the specific protocol which kind of reply the server sends to the user.

If we assume that the client does not blindly signs data and the server side knows that the public key belongs to the user, then the server side can conclude that the user owns the session key and the private key which corresponds to the account certificate. This can also be formalised with the help of the BAN logic [5,4]. We assume the following four preconditions:

1. $S \models \#(n)$: the server (S) believes that his challenge is fresh.
2. $S \models \overset{K_j}{\mapsto} J$: the server believes that K_j is the public key of the user J.
3. $S \triangleleft J \overset{K_{js}}{\longleftrightarrow} S$: the server has once seen the session key.
4. $S \triangleleft n$: the server has once seen the challenge he has generated.

Finally, we need to model the idealized response from the client to the server. We skipped to model the other messages, since they are not needed for the claim we want to show. Please note, we have skipped the outer encryption with the session key, it does not add any more security. The idealized version of the response is:

$$J \rightarrow S : \{H(J \overset{K_{js}}{\longleftrightarrow} S, n)\}K_j^{-1} \tag{1}$$

By applying the postulates of the BAN logic we can show $S \models J \models J \overset{K_{js}}{\longleftrightarrow} S$:

$$\dfrac{\dfrac{\dfrac{S \models \overset{K_j}{\mapsto} J, S \triangleleft \{H(J \overset{K_{js}}{\longleftrightarrow} S, n)\}K_j^{-1}}{S \models J \sim H(J \overset{K_{js}}{\longleftrightarrow} S, n)}, S \triangleleft J \overset{K_{js}}{\longleftrightarrow} S, S \triangleleft n}{S \models J \sim (J \overset{K_{js}}{\longleftrightarrow} S, n)}, \dfrac{S \models \#(n)}{S \models \#(J \overset{K_{js}S}{\longleftrightarrow} S, n)}}{\dfrac{S \models J \models (J \overset{K_{js}S}{\longleftrightarrow} S, n)}{S \models J \models J \overset{K_{js}}{\longleftrightarrow} S}}$$

Basically, this is the condition we want to achieve with our modification. Now the server side can assume that the client believes in the session key. Certainly, the result is only valid if prior server authentication protocol results in the assumptions 2 and 3.

6 Conclusion

AN.ON's payment concept utilizes the property of a cascade, every mix in a cascade transfers the same amount of traffic, to establish their prepaid payment concept. In this paper we analysed the most important cryptographic protocols of this interesting approach. The analysis of the protocols showed that they contain several weaknesses which need to be addressed to provide a fair service. In

addition we showed some practical implications of the weaknesses. For instance we showed that an accounting instance can abuse its position to consume some portion of the traffic the user has bought on different cascades. Beside this we also propose a solution which fixes the encountered weaknesses of the protocol.

Acknowledgements

We want to thank Rolf Wendolsky and the JonDos GmbH for their help to retrieve the current protocol version and Stig Mjølsnes for his PhD-course in *Cryptographic Protocols and Their Applications* as well as his fruitful comments.

References

1. Androulaki, E., Raykova, M., Srivatsan, S., Stavrou, A., Bellovin, S.M.: Par: Payment for anonymous routing. In: Borisov, N., Goldberg, I. (eds.) PETS 2008. LNCS, vol. 5134, pp. 219–236. Springer, Heidelberg (2008)
2. Berthold, O., Federrath, H., Köpsell, S.: Web MIXes: A system for anonymous and unobservable Internet access. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 115–129. Springer, Heidelberg (2001)
3. Berthold, O., Pfitzmann, A., Standtke, R.: The disadvantages of free mix routes and how to overcome them. In: Federrath, H. (ed.) Designing Privacy Enhancing Technologies. LNCS, vol. 2009, pp. 30–45. Springer, Heidelberg (2001)
4. Burrows, M., Abadi, M., Needham, R.: A logic of authentication. Technical Report 39, Digital Equipment Corporation Systems Research Center, Palo Alto, Calif. (February 1989) (Revised on February 22, 1990)
5. Burrows, M., Abadi, M., Needham, R.: A logic of authentication, vol. 8, pp. 18–36. ACM, New York (1990)
6. Chaum, D.: Untraceable electronic mail, return addresses, and digital pseudonyms. Communications of the ACM 4(2), 84–88 (1981)
7. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The second-generation onion router. In: Proceedings of the 13th USENIX Security Symposium (2004)
8. Müller, A.: Entwurf und Implementierung einer Zahlungsfunktion für einen Mix-basierten Anonymisierungsdienst unter Berücksichtigung mehrseitiger Sicherheitsanforderungen. Master's thesis, TU Dresden, Germany (2002)
9. Pfitzmann, A., Hansen, M.: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management - a consolidated proposal for terminology, v0.31 (February 2008)
10. Wendolsky, R.: A volume-based accounting system for fixed-route mix cascade systems. In: Bamberger Beiträge zur Wirtschaftsinformatik und angewandten Informatik, February 2008, pp. 26–33 (2008)

Formal Analysis of the Estonian Mobile-ID Protocol

Peeter Laud[1,2] and Meelis Roos[1,2]

[1] Cybernetica AS
[2] Tartu University, Institute of Computer Science

Abstract. In this paper, we report the results of the formal analysis performed on the Estonian Mobile-ID protocol (deployed since 2008), allowing citizens and permanent residents of Estonia to authenticate themselves and issue digital signatures with the help of a signature-capable SIM-card inside their mobile phone. We analyze the resiliency of the protocol to network attacks under various threat models (compromised infrastructure, client application, etc., confusing user interface) and give suggestions for improvement.

1 Introduction

Since 2002, Estonia has issued chipped ID cards to its citizens and permanent residents. The card has been integrated into a national public-key infrastructure. Upon the initialization of a new ID card for the user U, two RSA keypairs are loaded into it. The card is capable of performing modular exponentiations with the secret exponents of those keypairs. During initialization, certificates binding the public keys to the user U are also issued and stored on the card (as well as in a public database). The certificates are issued by a certification authority (CA) in the list of state-recognized CAs. The intended uses for the secret keys (as recorded in certificates) are identification (for the first keypair) and signing (for the second keypair).

The identification functionality of the card can be used when accessing public web-sites. When the user has directed his client application (usually a web browser) to access a server over a secured connection, the two will perform a TLS handshake [10] during which both the server and the client are authenticated. During the protocol, the client has to sign a message, a hash of which is handed over to the ID card in a smartcard reader connected to client's computer. The card will apply the RSA exponentiation to this hash, using the secret exponent in the first RSA keypair. The result is handed back to the client application which includes it in a protocol message. To activate the card's signing functionality, a PIN (consisting of four or more decimal digits) has to be given to it (different PINs for different keypairs). The PIN is entered either from the computer keyboard or the PIN-pad integrated with the card reader. In the first case, the PIN is handled by the client application and given to the card together with the hash to be signed. The card locks up after a couple consequtive incorrect enterings of the PIN.

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 271–286, 2009.

Since 2007, Eesti Mobiiltelefon (the largest Estonian mobile operator) in cooperation with Sertifitseerimiskeskus AS (the only state-recognized CA in Estonia) have issued mobile SIM cards with the same functionality [15]. Later that year, they were joined by the Lithuanian mobile operator Omnitel [20]. Similarly, RSA keypairs are loaded into those cards and the public keys are issued certificates binding them with users. The SIM card can compute signatures on users' behalf after being given a PIN which is entered from the keypad of the mobile phone. The mobile ID thus reduces the threats related to handing over one's PIN to a possibly trojaned computer. Trojan horse attacks on mobile phones are as of now only a negligible part of the malware market [13], although should their number and impact increase, the conclusions made in this paper must be reconsidered. Another claimed advantage of mobile ID is convenience — no smartcard reader is necessary [16].

At the same time, client authentication in the Mobile-ID protocol uses a much more complex protocol than the TLS handshake, and involves many more parties. This raises a number of interesting security questions. The goal of the research reported in this paper was to formalize the Mobile-ID protocol in the protocol checker ProVerif [7] and use it to explore what happens if various parts of the system are acting differently than expected. We also have tested the implementation of central parts of the protocol; this paper shows how to formally model the possible weaknesses we found. After reporting the results of this exploration, we also suggest modifications for the protocol to make it more secure under certain attacks.

A general security analysis of Mobile-ID has been performed previously [21]. This analysis was considerably broader in scope than the one reported in this paper; it considered not just the network attacks, but also legal and human issues and risks related to the failure of technical components. The conclusion of the analysis was that generally, the risks associated with Mobile ID are the same as the risks of using the ID card. There are some additional risks related to the necessity to trust the extra infrastructure used in the Mobile ID protocols. No formal analysis of the protocols was presented in [21].

In this paper we first describe the Mobile-ID authentication protocol and base security assumptions (honesty of certain parties and security of certain channels) for it. The base security assumptions describe the situation where only parties that are normally considered to be dishonest can be dishonest. As next we describe how we have formalized the Mobile-ID protocol in ProVerif. In the next section we describe the results of our analysis. We have analyzed the protocol under base security assumptions, as well as several different, stronger assumptions where we have allowed certain parties or connections to be under adversarial control. We finish the paper with suggestions for improving the protocol, as well as general conclusions.

2 The Mobile-ID Protocol

The Mobile-ID protocol [4] involves the following parties:

- The user U that wants to access some service requiring authentication.
- The phone P of that user, as well as the SIM card inside it. Although technically two different units, we model them as a single one. The SIM card knows the secret signing key sk_U, the corresponding public key pk_U of which is bound to the identity of U by the certificate cert_U.
- The client application C of that user, typically a computer running a web browser. The client application is used to actually access the service.
- The server S that the user wants to connect to. It has a secret key sk_S which public counterpart pk_S is bound to the name S by the certificate cert_S. With the help of sk_S, the server can participate in a TLS handshake as a server.
- The mobile operator O that has issued the SIM-card of the phone P.
- The DigiDocService D [4]. This is a central party of the protocol meditating the authentication process and forwarding the messages to right parties. The DigiDocService has a secret key sk_D allowing it to participate in a TLS handshake as a server. The certificate cert_D binds the corresponding public key pk_D to the identity of D.

The parties above actively take part of a protocol session. Besides them, there is also the certificate authority CA issuing the certificates. Also, there are means (OCSP) to verify the status of a certificate [19].

The Mobile-ID protocol [4] is depicted in Fig. 1. A protocol session is initiated by the user U deciding to contact the server S and informing the client application C about this choice. The client application locates the certificate cert_S of the server and initiates a TLS handshake with it. During the handshake, the server is authenticated to the client, but not vice versa. The resulting TLS tunnel is used to communicate the rest of the messages between C and S. To authenticate the user, C sends to S the name U (which also determines the phone number P). The server S then initiates a TLS handshake with D, again resulting in the secure identification of D, but not S. Again, the TLS tunnel is used to encapsulate the messages between S and D. The server S sends to D the names U and P, identifying the user. Additionally, S generates a 10-byte challenge r_1 (part of the challenge signed with sk_U) and sends it to D, too. Optionally, r_1 may be empty. Also, S sends to D something that identifies itself: $\tilde{S} = (S, m)$ where m is an additional message that will be displayed to the user on the screen of the mobile phone. Both S and D will then locate the certificate cert_U of the user.

The DigiDocService will generate a random nonce r_2. The phone of the user is supposed to sign the concatenation of r_1 and r_2 with the key sk_U, where pk_U is included in cert_U. DigiDocService forwards both $\tilde{S}$ and $r_1 \| r_2$ to the phone P via the mobile operator O. The communication between D and O is protected by a VPN solution. The communication between O and P is through SMS-s, and is protected by encrypting the messages between O and P with the symmetric key $\tilde{K}_P$ known only to themselves. DigiDocService also computes CC_1 as the *control code* of $r_1 \| r_2$. The control code consists of four decimal digits. The control code CC_1 is forwarded to the client application C that displays it to the user U. The phone P also computes the control code of $r_1 \| r_2$ and displays it to the user, along with the name of the server S and the accompanying message m. The

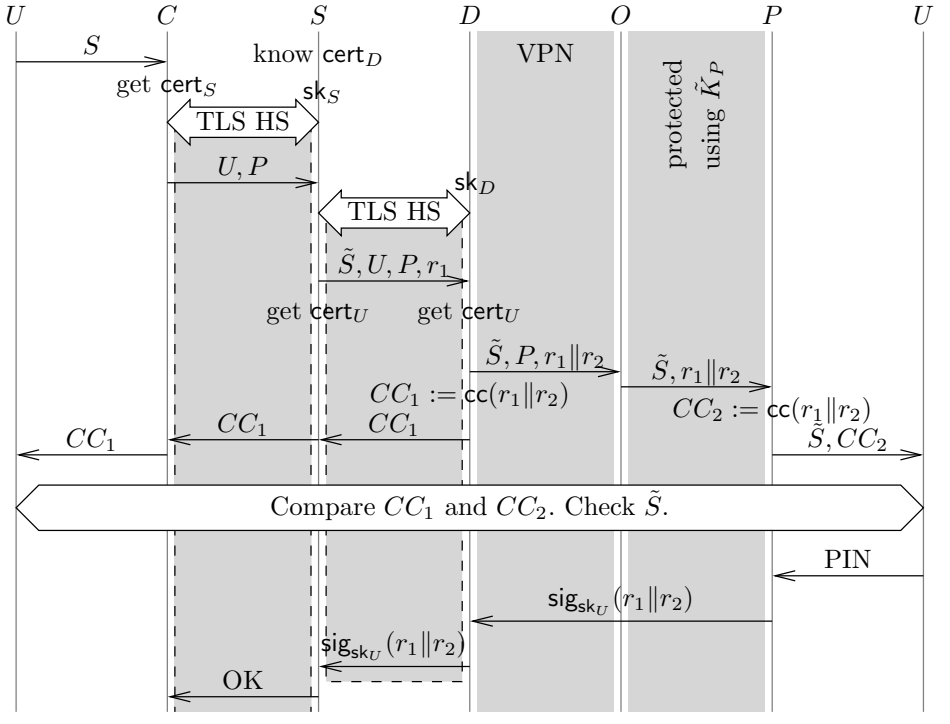


Fig. 1. Mobile-ID protocol

user checks that the control codes displayed by the client application and the phone are equal. The user also checks that the name of the server displayed by the phone is equal to the server he wanted to access, and the message m makes sense. If the checks succeed then the user instructs the phone that it is OK to sign the challenge, and provides the PIN for identification. The phone, receiving the PIN, signs the challenge $r_1 \| r_2$ and forwards it to D via O. DigiDocService D verifies that signature. If the signature verification is successful, and r_1 is not empty, then the signature is forwarded to S that also checks it, and upon success deems U to be authenticated. If r_1 is empty (i.e., S did not provide a challenge) then D does not forward the successfully verified signature to S, but only sends the confirmation that the verification succeeded. Again, S considers U to be authenticated. The TLS tunnel between C and S is then used for the regular communication.

2.1 Base Security Model

There are several entities, with several channels between them. Certain of those may be controlled by the adversary. In our "base" model we make the following assumptions about the security of various channels and entities:

1. There are several users and servers, some of them may be under adversarial control.
2. There is a single DigiDocService and mobile operator. They are honest. The channel between them is secure. In reality, these parties are relatively large organizations under significant public scrutiny. Still, we will relax this assumption in certain models.
3. The phones and client applications are under the control of their respective users.
4. The channel between a user and his client application is secure. So is the channel between a user and his phone. This is a reasonable assumption (for the base model, where we do not consider trojaned devices) as these channels are realized by the keyboards and screens of computers and phones.

The basic security property that we are interested in is the secrecy of the TLS keys agreed by honest clients and servers. We are also interested in correspondence properties: if a server S thinks that it talks to a client controlled by the user U using the key K and U is honest, then U must also think that it talks to the server S in a session where his client application C is using the key K (*integrity for servers*). Similar property must hold if we swap the user and the server (*integrity for clients*). Note that integrity for clients is derived directly from the properties of TLS because the server is authenticated during TLS handshake. TLS is a thoroughly researched protocol [12] and we know that it provides integrity for clients. Therefore we will subsequently only be concerned with the integrity for servers.

3 Formalization in ProVerif

ProVerif [7] is a protocol analyzer in the formal (or: Dolev-Yao) model [11]. To apply it to a protocol, it has to be formalized in a language reminiscent to the applied π-calculus [3]. In this calculus, messages are represented by *formal expressions* made up of free names and expression constructors, the set of constructors is fixed for a protocol. The process is expressed in a language containing primitives for sending and receiving messages (the channel has to be specified, too; it is a name), generating new names (modeling the generation of new keys, nonces, etc.), constructing and deconstructing messages, branching, sequential and parallel composition, and replication. The input language of ProVerif also contains means to specify the security properties (both secrecy and correspondence properties, as well as various process equivalences that we are not using here). ProVerif is a mature tool, having been used to check the security of various key-exchange [7,2], authentication [8], fair exchange [1], secure storage [9], electronic voting [17,5], etc. protocols. The tool is capable of modeling different cryptographic primitives, including Diffie-Hellman key exchange [2] and non-interactive zero-knowledge [6].

Our model of the Mobile-ID protocol, following the base security model consists of the following parts.

TLS handshake. We follow the modeling by Tankink and Vullers [22]. They have verified that the TLS handshake provides integrity for clients. The TLS handshake is used as a subprotocol in two different places of the Mobile-ID protocol. We use the trick described by Haack [14] to include TLS handshake as a subprotocol, without duplicating its code.

Certification. Instead of including a full-fledged CA process in our model, giving signatures to certificate requests, we have included a private expression constructor cert, such that $\mathsf{cert}(X, \mathsf{pk}_X)$ represents that pk_X is the public key of X. The privacy of the constructor means that the adversary cannot use it to construct new expressions. On the other hand, we have included destructors that the adversary can use and read both components of a cert-message. The honest users, servers and DigiDocService generate their keys and publish the corresponding certificates at the beginning of their processes. To give certificates to dishonest users and servers, we add a (replicated) process that takes a public key pk as an input, generates a new name n and outputs $\mathsf{cert}(n, pk)$ on a public channel. It is important that the name is newly generated, otherwise the adversary could issue new certificates to honest parties.

Phone registration. The binding of the key $\tilde{K}_P$ to the phone P is handled similarly — there is a private binary constructor $\mathsf{phonereg}$ representing the binding of a key to a user's phone. There are also destructors to read both components of a $\mathsf{phonereg}$-message, but only the one giving the name of the user is public (i.e., can be invoked by the adversary). Binding a key to the phone of a dishonest user is handled similarly to certification. Actually, the process described in the previous paragraph is extended to also output a $\mathsf{phonereg}$-message.

User, client application and phone. We model these parties as a single process (with several replicated subprocesses). The process first generates a new name and keys for signing and mobile communication and publishes the certificates for them. The process will then split into several parallel subprocesses, each of them replicated. These subprocesses are described below.

One of the subprocesses models the client application in one protocol session. It receives the name of the server to connect to (from the user subprocess), runs the TLS handshake with the server, verifying server's identity in that process, receives the control code from the server through the established TLS-tunnel and sends it to the user subprocess. The channel between the user and client subprocesses is a secure one; its name is generated before the parallel subprocesses start.

Another subprocess models both the user and the phone during one protocol session. It tells the client application to start connecting to a server (the name of the server is received from the adversary), gets back the control code from it, and also gets the challenge to be signed and information identifying the server from the network, encrypted with the key for mobile communication. The process verifies whether the control code from the client application matches the control code of the challenge (also checks the identity of the server). If the check succeeds, it signs the challenge and sends it back, encrypted.

Third subprocess is used to indicate that this user is honest. It sends the name of the user on a private channel (a free name that the adversary does not have access to).

Other parties. The processes modeling a server, the DigiDocService, and the mobile operator are straightforward. The server process first generates the name and the key of the server, publishes the certificate cert_S and then runs an unbounded number of processes implementing the server part of the Mobile-ID protocol. The name of the DigiDocService is globally known, hence the DigiDocService process starts by generating only the key and publishing the certificate for it. We use a private channel (a free name that the adversary cannot use) to model the VPN used for communication between the DigiDocService and mobile operator.

The whole system. The analyzed process consists of the parallel composition of the client process (replicated), server process (replicated), DigiDocService process, mobile operator process, processes for TLS handshake (replicated) and the process for issuing certificates for dishonest clients and servers.

Checking the control code. The control code consists of four decimal digits, hence collisions are easy to construct. It would be wrong to model the control code just by a message constructor with no additional equations as that would hide the very real possibility of control code collisions. In our model, we still introduce the constructor cc, such that $\mathsf{cc}(r)$ is the control code corresponding to r, but instead of using equality of terms to check the control code in the user process, we have introduced a binary predicate $\mathsf{TestCCEq}$ (ProVerif supports such introduction of predicates). The invocation of $\mathsf{TestCCEq}(r, c)$ is supposed to return true if c is the control code corresponding to r (recall that the user process receives the control code from the client process and the challenge from the mobile operator). Our model contains the clause $\mathsf{TestCCEq}(x, \mathsf{cc}(x))$. Additionally, it contains the clauses for modeling that given c, the adversary can construct messages of certain shapes whose control code is c. The shapes of these clauses depend on the attacks that the adversary may want to perform. In the weakest case (the adversary can find preimages of a given control code, but cannot control the shape of the preimage) the clause is $\mathsf{TestCCEq}(\mathsf{invcc}(x), x)$, where invcc is a new message constructor. We consider stronger cases when we study different security models. Our model does not consider the possibility that two control codes might be equal by chance. An honest DigiDocService can easily ensure that challenges with equal control codes are not awaiting signatures of the same user at the same time.

Security properties. We are interested in two security properties — the secrecy of the keys of the TLS-tunnel between honest clients and servers, and the authentication of users to servers. Our model in ProVerif contains queries for verifying these two properties. For verifying the secrecy of keys, we have introduced a private free name M. The server process encrypts M with the keys of

the TLS tunnel at the end of each protocol round (at the bottom of Fig. 1) and makes the resulting ciphertext public. The query asks ProVerif whether M is still secret.

For the correspondence properties we use the *events*. An event E is a program statement that is semantically equivalent to a no-operation, but records that the program point containing event E has been passed (E has happened). ProVerif can answer queries of the form "if event E_2 has happened, then must the event E_1 also have happened?" In our model, we add an event $\mathsf{ServerEnd}(U, S, k)$, where k is the key for the TLS-tunnel between S and C, to the end of the server process, after it has accepted that user U has been authenticated. We also add an event $\mathsf{UserEnd}(U, S, k)$ at the point where the user has completed all of his steps to be accepted by the server — at the point where the user must enter his PIN to the phone. The user process does not normally have the key k. Therefore the client process will send k to the user process, too. The query asks whether the event $\mathsf{ServerEnd}(U, S, k)$ implies $\mathsf{UserEnd}(U, S, k)$.

Both properties are easily invalidated if the user is dishonest. Hence the server performs the actions for both properties (publishing the encryption of M, and performing the event $\mathsf{ServerEnd}$) only if the user is honest. The user is honest if the server can receive his name over the private channel for honest user names (see the description of user, client application and phone processes above).

4 Verification Results

The Mobile-ID protocol, as we have modeled it in Sec. 3, following the security model of Sec. 2.1 is deemed secure by ProVerif — the correspondence property holds and the message M cannot be found by the attacker. Still, this only reflects an in some sense "ideal" situation. Let us now consider the protocol where certain things go wrong with respect to the base security model.

4.1 Attacker Controls DigiDocService

DigiDocService is a mediator of messages, helping the protocol to proceed. It would be unnatural if the *security* of the protocol depended on its actions. It is straightforward to model DigiDocService being under adversarial control — we make public its secret key sk_D, as well as the channel between it and the mobile operator.

Being under adversarial control, the DigiDocService is expected to look for collisions in control codes for challenges. As it can fix the second half of the challenge, we expect that DigiDocService desires to solve the problems of the following form: given c and r_1, find r_2 so that $\mathsf{cc}(r_1 \| r_2) = c$. This is a reasonably solvable problem, and we add a clause to the definition of $\mathsf{TestCCEq}$ stating its solvability. Namely, we introduce a binary message constructor postc and state that $\mathsf{TestCCEq}((r_1 \| \mathsf{postc}(c, r_1)), c)$ holds.

ProVerif finds an attack violating both security properties. This attack should not even be so surprising, because the construction of the signature $\mathsf{sig}_{\mathsf{sk}_U}(r_1 \| r_2)$

violates certain prudency criteria for the construction of cryptographic protocols [18, Ch. 11]. If the adversary wants to masquerade as U to a server S then it waits until U wants to contact some server S', and proceeds as follows:

- A contacts S and performs the TLS handshake with it. At the same time, U is performing the TLS handshake with S'.
- A identifies itself to S as U. S contacts DigiDocService D (under control of A), performs the TLS handshake with it and forwards it the name U (and P) together with its own name $\tilde{S}$ (including the additional message m) and its half of the challenge r_1. At the same time, U identifies itself to S', which also performs the TLS handshake with D and forwards to it the names U and P, its own name $\tilde{S}'$ and the half of the challenge r'_1.
- The adversary (as D) constructs r_2 and r'_2 so, that $\mathsf{cc}(r_1 \| r_2) = \mathsf{cc}(r'_1 \| r'_2)$. Let c be this control code. The adversary (as D) sends c back to S and S'. It also sends $\tilde{S}'$, P, and $r_1 \| r_2$ to the mobile operator, which forwards them to P.
- The server S' sends c back to U. The phone P shows $\tilde{S}'$ to the user U. The user agrees that it intended to contact S'. The phone also shows the control code of $r_1 \| r_2$ to the user. This happens to equal c.
- The user enters his PIN to the phone and the phone signs $r_1 \| r_2$. This signature is forwarded to S (via the mobile operator and the adversary posing as DigiDocService), and S accepts the connection with A as coming from U.

Note that here the adversary only controls D, and not any other parties. Therefore this is a very powerful attack.

The attack succeeds even if the collisions for control codes were impossible to construct. Impossibility of collisions means that the control codes must be so much longer (at least 40-50 decimal places) that it would seriously degrade the usability of the system. In this case, the attack is possible if S' is under adversarial control. Compared to the described attack, we now just take $r'_1 = r_1$ and $r'_2 = r_2$, and we do not have to look for collisions.

A prudent protocol design guideline says that when constructing a signature, let the name of the intended verifier be a part of the signed message. This guideline has not been followed in the design of the Mobile-ID protocol. We could modify the protocol by letting P include the name S under the signature it generates, and subsequently verifying that a correct name has been included.

This change still does not fix the protocol, but now the original attack succeeds only if $S = S'$. In other words, the user U initiates one session with the server S and the attacker initiates a different session at the same time. The adversary (in the role of D) again finds r_2 and r'_2 so that there is a collision in the control codes. The modified protocol might be secure if a server does not allow the same alleged user to run two sessions in parallel. Unfortunately, we do not know of a simple means to model such restriction in ProVerif.

ProVerif deems the protocol secure if both modifications (no control code collisions and server name under signature) are made. To ensure that adversarially controlled DigiDocService does not generate control code collisions, the server itself should generate the whole challenge. In terms of Fig. 1, r_2 should be the

empty string and r_1 should be long enough to be unpredictable. The server must also make sure to not issue challenges with the same control code for parallel sessions of allegedly the same user. It goes without saying that the control code CC_1 sent to the user via his client application must be computed by the server, not the DigiDocService.

4.2 Attacker Partially Controls the Client

One of the goals of the Mobile-ID protocol was to reduce the effect that a compromised client machine might have on the security of authentication. Clearly, if the adversary has completely taken over the client computer, then it knows the keys for the TLS-tunnel between the client and the server and can listen and speak on behalf of the user. Still, even in this case the adversary cannot contact a server S on behalf of a user U while the user U remains completely passive: ProVerif claims that even in this case the event $\mathsf{ServerEnd}(U, S)$ implies the event $\mathsf{UserEnd}(U, S)$ (note that we do not include the key for the TLS-tunnel in the arguments of these events) and moreover, the correspondence is injective: for each action of the user (instructing the phone to sign a challenge), the adversary can start at most one session with the server. To model in ProVerif that the adversary controls the client machine, we make public the secret that this process uses: the channel between the client application and the user.

A keylogger does not have to take over the whole machine in order to cause harm (record the PIN of the ID card). A similarly interesting case for the Mobile-ID protocol is, when the malware has not taken over the whole machine, but can influence how the control code is shown to the user. This case models malware that can redraw the screen. This change is simple to model — in Fig. 1, the value CC_1 is received by the user U not from C, but from public network.

ProVerif finds, that if the adversary controls the value of CC_1 as presented to U, then the protocol is insecure. If the adversary A wants to masquerade as the user U to a server S, then it proceeds as follows.

- Wait until U himself contacts S. As we explained in the first paragraph, it is impossible to initiate a session (as U) with S, unless U himself also wants to contact S.
- Start a session with S, claiming to be U. Let both sessions proceed to the point where DigiDocService has constructed control codes c (for U) and c' (for A) and sent them to S and to P.
- The adversary makes sure that the challenge with the control code c' reaches P first. This can be achieved with right timing. The adversary also makes sure that the second control code is not shown to the user before it has accepted c'. By our experimentations with DigiDocService[1], this condition trivially holds — a mobile phone does not hint of the existence of a second incoming control code before the user has taken action on the first one.
- The adversary receives c', the client application receives c. The dishonest client application now contacts the adversary, learns c', and shows it to the

[1] http://www.sk.ee/DigiDocService/DigiDocService_2_3.wsdl

user instead of c. The user confirms that the client application and the phone show the same control code c' and instructs the phone to sign the challenge. This challenge corresponds to the session between A and S.

The attack should be avoidable if the server does not start several sessions with the same user in parallel. Indeed, if a session has ended and the user has generated the event $\mathsf{UserEnd}(U, S)$ then the server has also generated the event $\mathsf{ServerEnd}(U, S)$ and because of injective correspondence, this event $\mathsf{UserEnd}(U, S)$ cannot be used to match a different $\mathsf{ServerEnd}(U, S)$ taking place later (presumably because of adversarial activity).

4.3 User Confused Regarding the Server Names

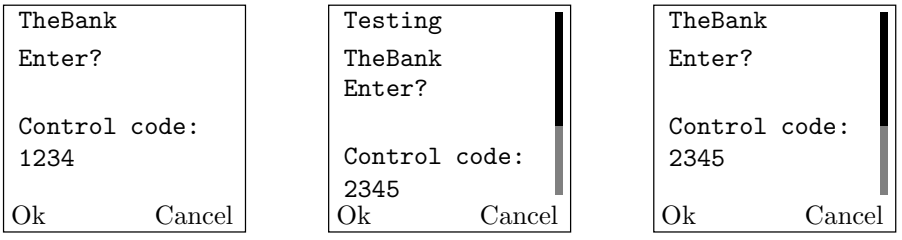
An important class of attacks are *semantic attacks* where the adversary tries to convince the user that a wrong statement holds. One example of such attacks are the phishing web-sites masquerading as legitimate ones. They typically have names similar to the one they are trying to masquerade. Authentication using an ID card is relatively immune to such attacks — while an attacker can obviously make the user connect to a fake server (if the user does not notice its fakeness), this cannot be used to masquerade the user to a legitimate server.

We studied how well the Mobile-ID protocol fared against such attacks. We assumed that there is an adversarially controlled server S' that is hard to distinguish from a legitimate server S. It turned out, that a classical man-in-the-middle attack is possible, allowing the adversary to masquerade as U to S. In this attack, U connects to S' thinking it is S, while the adversary (masquerading as U) connects to S. The server S contacts D and the phone of the user receives the challenge and the information $\tilde{S}$ identifying the server. The control code is also sent from D back to S, which forwards it to A, which forwards it to U as S'. The phone shows S as the name of the server, the user is connected to S', but we assume that he does not notice the difference. The control codes shown by the phone and the client application are the same. The user hence tells the phone to sign the challenge and S will accept A as U. The attack works even if the name of the server is included in the signed message.

4.4 Server Chooses the Control Code

When server S contacts the DigiDocService D, it sends it not only the name S, but also the up to 40 characters long message m; both S and m will be shown to the user on his phone. A typical picture of the phone screen is shown in Fig. 2a. Here S equals "TheBank" and m equals "Enter?". The next lines have been produced by software running inside the phone (actually, inside the SIM-card). They inform the user that the control code of the challenge is 1234.

The possible values for S have been enumerated by the DigiDocService D and D also weakly authenticates S by its IP-address. The server named "Testing" may connect from any IP-address. We have experimented with DigiDocService and various values of m, using this server identity. We have discovered that if m



(a) (b) (c)

Fig. 2. Typical screen for comparing the control code

contains embedded newlines, then these are shown as line breaks on the phone screen. E.g., if m equals "TheBank\nEnter?\n\nControl code:\n2345\n" then the phone screen (for certain models) will look as shown in Fig. 2b. Here the entire message shown by the phone has not fit on the screen and a scroll bar has appeared on the right-hand side. If we scroll down, we see the control code that has been computed by the phone itself and may be different from 2345. Depending on the model of the phone, this scroll bar may be rather hard to notice.

We believe that with IP-spoofing, we can cause the picture in Fig. 2c to appear on the phone screen. Also, if the adversary controls either TheBank or the DigiDocService (as we have argued before, this case has to be analyzed, too) then it is straightforward to make this picture appear, as the DigiDocService can control which S and m are sent to the phone. Hence we conclude that a malicious server or DigiDocService can (under certain circumstances) control which four-digit number is shown to the user as the control code.

We have modeled this scenario with ProVerif. The necessary modifications involve several parts of the model, as the fake control code is inserted at the server, and then travels to DigiDocService, mobile operator, phone, and the user. At the same time, the changes are rather straightforward.

We have considered the case where the DigiDocService is honest, but a malicious server is capable of entering a fake control code, eventually shown to the user by the phone. Somewhat surprisingly, the protocol is still deemed secure. One may conjecture that the user might not need to perform the equality check of control codes at all. Of course, this is not so; there exists a straightforward parallel attack: both U and A connect to S, both claim to be U, the challenge for A's session is the first to arrive at U's phone, U does not check the equality of control codes and causes this challenge to be signed. Note that U still checks that the phone shows the name of the server S (the opposite is considered in Sec. 4.3). The reason why there is no attack if a malicious server can pick a fake control code, is that to use this capability, the attacker has to set up a malicious server S' whose name will be shown to and rejected by the user.

In Sec. 4.1 we gave some suggestions to handle a dishonest DigiDocService. We have checked the security of our modifications (server name under signature, challenge generated entirely by the server, no control code collisions) if a server

or the DigiDocService can also choose the control code that the phone shows. ProVerif gives us an attack. The attack is similar to the attack presented in Sec. 4.2. The only difference is that now the attacker changes the control code shown by the phone, not the control code shown by the client application. Again, the attack should not work if the server does not start several sessions with the same user in parallel.

5 Proposed Improvements

We suggest the following modifications to the Mobile-ID protocol to increase its security:

- When the phone signs the challenge $r_1 \| r_2$ for the server S, the signed message should not be $\mathsf{sig}_{\mathsf{sk}_U}(r_1 \| r_2)$, but $\mathsf{sig}_{\mathsf{sk}_U}(r_1 \| r_2, S)$. The presence of S under the signature must be checked by parties receiving that signature.
- r_2 should be a constant, most naturally the empty string. The control code CC_1 should be computed by S, not D. The server S should generate the challenges r_1 in such a way that the sessions of the same alleged user U have challenges with different control codes.

We also suggest that the user interface of the phone should be modified in a way that the control code is always in the same place at the phone screen, and always visible when the message is first shown to the user. This can be achieved by showing the control code before the message m, or by appropriately filtering m. Users should also be educated to look for the control code in a certain place.

6 Summary

Above, we have considered attackers of various strength. They all had the ability to initate protocol sessions; they controlled certain users and servers and had no access to the phones of the users. Their strength varied along the following dimensions:

- $\mathbf{d}_1$ — control over the DigiDocService or mobile operator (possible values: 0 — no control, 1 — full control);
- $\mathbf{d}_2$ — control over the client application (0 — no control, 1 — can change displayed CC, 2 — full control);
- $\mathbf{d}_3$ — ability to confuse the user about server names (0 — no, 1 — yes);
- $\mathbf{d}_4$ — ability for a compromised server to pick the control code shown on phone screen (0 — no, 1 — yes).

We see that the abilities of attackers may include the corruption of any party in Fig. 1, except the phone P, taking the advantage of the user interface issues in both C and P, and phishing attacks. We have not considered the attacker gaining significant control over the phone. Indeed, any reasonable attack model would allow the adversary to learn the PIN entered from the keypad of the phone; the

knowledge of the PIN gives the adversary full capabilities of masquerading as the user U. We have also not considered the user's failure to compare the control codes shown by the client application and the phone, but this is subsumed by the dimension $\mathbf{d}_2$. To summarize, we believe that we have not left any significant attack vectors without consideration.

We have proposed two mutually independent protocol modifications. These propositions introduce dimensions on whether they have been taken into account.

- $\mathbf{d}_5$ — is the name of the server included under the signature of the challenge? (0 — yes, 1 — no)
- $\mathbf{d}_6$ — is the half r_2 of the challenge empty? (0 — yes, 1 — no)

Another dimension is introduced by an honest server's behaviour when allegedly the same user attempts to authenticate himself several times in parallel:

- $\mathbf{d}_7$ — does S allow parallel sessions with the same U? (0 — no, 1 — yes, but picks challenges with different control codes, 2 — yes) Note that $\mathbf{d}_7 = 0$ means that the server lets a session with a user U to time out before agreeing to participate in a different session with U. This may make denial-of-service attacks too simple.

Let $\mathbf{L}_j^i$ denote the predicate $\mathbf{d}_j \leq i$. Our analysis shows that the security properties described in the end of Sec. 3 hold if

$$(\mathbf{L}_1^0 \vee (\mathbf{L}_5^0 \wedge \mathbf{L}_6^0 \wedge \mathbf{L}_7^1)) \wedge \mathbf{L}_2^1 \wedge (\mathbf{L}_2^0 \vee \mathbf{L}_7^0) \wedge \mathbf{L}_3^0 \wedge (\mathbf{L}_4^0 \vee \mathbf{L}_1^0 \vee (\mathbf{L}_5^0 \wedge \mathbf{L}_6^0 \wedge \mathbf{L}_7^0)) \ .$$

Indeed, the justification for each of the conjuncts is the following:

- We showed in Sec. 4.1 that an adversarially controlled DigiDocService is capable of breaking the protocol unless the modifications stated in Sec. 5 were introduced.
- If the adversary has full control over the client application, then it can take over the connection between C and S.
- In Sec. 4.2 we showed that if the adversary can change the control code shown to the user by the client application, then there exist an attack that requires parallel sessions with the same server.
- In Sec. 4.3 we argued that the mobile-ID protocol does not protect against phishing attacks, even if we implement the modifications in Sec. 5.
- In Sec. 4.4 we showed that the capability for a malicious server to choose the control code displayed by the phone is not enough for breaking the security properties, but if the DigiDocService is also under adversarial control then the modifications of Sec. 5 no longer suffice to preserve the security, but parallel sessions between the same alleged user and server must be ruled out. Hence we suggested in Sec. 5 to make sure that $\mathbf{L}_4^0$ holds.

7 Conclusions

We have analyzed the security of the Mobile-ID protocol introduced by an Estonian CA and Estonian and Lithuanian mobile operators. We have discovered

some weaknesses in the protocol which manifest under strong adversarial models. Despite these weaknesses, we believe that the usage of the protocol can continue in the immediate future. Indeed, we believe that the attack vectors included in those adversarial models either will not materialize in the immediate future, or their materialization would allow attacks of similar success against other authentication methods, sometimes including ID card based methods. Still, the weaknesses should nevertheless be fixed with high priority.

Our analysis also shows that compared to other methods of authentication (passwords, one-time passwords, PIN-calculators), Mobile-ID does not offer significant protection against user errors or weaknesses of the client application. We conclude that it is too premature to state that *modulo* negligible risks, Mobile-ID is at least as secure as authentication with ID card [21].

Acknowledgments

This research has been supported by Estonian Science Foundation, grant #6944, by the European Regional Development Fund through the Estonian Center of Excellence in Computer Science, EXCS, and by Sampo Pank. We are grateful to Dan Bogdanov, Ilja Livenson, and Mari Seeba for fruitful discussions.

References

1. Abadi, M., Blanchet, B.: Computer-Assisted Verification of a Protocol for Certified Email. In: Cousot, R. (ed.) SAS 2003. LNCS, vol. 2694, pp. 316–335. Springer, Heidelberg (2003)
2. Abadi, M., Blanchet, B., Fournet, C.: Just Fast Keying in the Pi Calculus. In: Schmidt, D. (ed.) ESOP 2004. LNCS, vol. 2986, pp. 340–354. Springer, Heidelberg (2004)
3. Abadi, M., Fournet, C.: Mobile values, new names, and secure communication. In: 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL), London, UK, January 2001, pp. 104–115 (2001)
4. AS Sertifitseerimiskeskus. DigiDocService specification, v. 2.122, April 24 (2007), http://www.sk.ee/files/DigiDocService_spec_eng.pdf
5. Backes, M., Hritcu, C., Maffei, M.: Automated Verification of Remote Electronic Voting Protocols in the Applied Pi-Calculus. In: 21st IEEE Computer Security Foundations Symposium, CSF 2008, Pittsburgh, Pennsylvania, June 2008, pp. 195–209 (2008)
6. Backes, M., Maffei, M., Unruh, D.: Zero-Knowledge in the Applied Pi-calculus and Automated Verification of the Direct Anonymous Attestation Protocol. In: 2008 IEEE Symposium on Security and Privacy, May 2008, pp. 202–215 (2008)
7. Blanchet, B.: An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In: 14th IEEE Computer Security Foundations Workshop (CSFW-14), Cape Breton, Nova Scotia, Canada, June 2001, pp. 82–96 (2001)
8. Blanchet, B.: From Secrecy to Authenticity in Security Protocols. In: Hermenegildo, M.V., Puebla, G. (eds.) SAS 2002. LNCS, vol. 2477, pp. 342–359. Springer, Heidelberg (2002)

9. Blanchet, B., Chaudhuri, A.: Automated Formal Analysis of a Protocol for Secure File Sharing on Untrusted Storage. In: IEEE Symposium on Security and Privacy, Oakland, CA, May 2008, pp. 417–431 (2008)
10. Dierks, T., Rescorla, E.: The Transport Layer Security (TLS) Protocol, Version 1.1. IETF Network Working Group, RFC 4346 (April 2006)
11. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Transactions on Information Theory 29(2), 198–207 (1983)
12. Gajek, S., Manulis, M., Pereira, O., Sadeghi, A.-R., Schwenk, J.: Universally Composable Security Analysis of TLS. In: Baek, J., Bao, F., Chen, K., Lai, X. (eds.) ProvSec 2008. LNCS, vol. 5324, pp. 313–327. Springer, Heidelberg (2008)
13. Golovanov, S., Gostev, A., Maslennikov, D.: Kaspersky Security Bulletin 2008: Malware Evolution (January- June 2008), http://www.viruslist.com/en/analysis?pubid=204792034#9
14. Haack, C.: Verification of Security Protocols, ProVerif's Resolution Method, lecture slides (March 2008), http://www.cs.ru.nl/~chaack/teaching/2IF02-Spring08/
15. idBlog. EMT Launches the Mobiil-ID Service, May 2 (2007), http://www.id.ee/blog_en/?p=20
16. ID.ee. Mobile-ID main page, November 20 (2008), http://www.id.ee/10995
17. Kremer, S., Ryan, M.: Analysis of an Electronic Voting Protocol in the Applied Pi Calculus. In: Sagiv, M. (ed.) ESOP 2005. LNCS, vol. 3444, pp. 186–200. Springer, Heidelberg (2005)
18. Mao, W.: Modern Cryptography: Theory and Practice. Prentice Hall, Englewood Cliffs (2003)
19. Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. IETF Network Working Group, RFC 2560 (June 1999)
20. Šablinskas, R.: Summary of Mobile-ID launch in Lithuania. Minutes of the Baltic WPKI Forum Steering Committee, October 31 (2007), http://wpki.eu/Launch-of-mobile-ES-BalticWPKI.pdf
21. Security Analysis of Mobile ID (Summary, in Estonian). Ordered by Department of State Information Systems, fulfilled by Jaak Tepandi, July 11 (2008), http://www.riso.ee/et/files/MOBIIL-ID_kokkuvote_11-07-2008.pdf
22. Carst Tankink, Pim Vullers. Verification of the TLS Handshake protocol, May 20 (2008), http://www.cs.ru.nl/ chaack/teaching/2IF02-Spring08/tv-report.pdf

Generating In-Line Monitors for Rabin Automata

Hugues Chabot, Raphael Khoury, and Nadia Tawbi

Laval University, Department of Computer Science and Software Engineering,
Pavillon Adrien-Pouliot, 1065, avenue de la Medecine Quebec City, Canada
{hugues.chabot.1,raphael.khoury.1,Nadia.Tawbi}@ulaval.ca

Abstract. A promising solution to the problem of securing potentially malicious mobile code lies in the use of program monitors. Such monitors can be in-lined into an untrusted program to produce an instrumented code that provably satisfies the security policy. It is well known that enforcement mechanisms based on Schneider's security automata only enforce safety properties [1]. Yet subsequent studies show that a wider range of properties than those implemented so far could be enforced using monitors. In this paper, we present an approach to produce a model of an instrumented program from a security requirement represented by a Rabin automaton and a model of the program. Based on an a priori knowledge of the program behavior, this approach allows to enforce, in some cases, more than safety properties. We provide a theorem stating that a truncation enforcement mechanism considering only the set of possible executions of a specific program is strictly more powerful than a mechanism considering all the executions over an alphabet of actions.

Keywords: Computer Security, Dynamic Analysis, Monitoring Software Safety.

1 Introduction

Execution monitoring is an approach to code safety that seeks to allow an untrusted code to run safely by observing its execution and reacting if need be to prevent a potential violation of a user-supplied security policy. This method has many promising applications, particularly with respect to the safe use of mobile code.

Academic research on monitoring has generally focused on two questions. The first relates to the set of policies that can be enforced by monitors and the conditions under which this set could be extended. The second question deals with the way to in-line a monitor into an untrusted or potentially malicious program in order to produce a new instrumented program that provably respects the desired security policy.

While studies on security policy enforcement mechanisms show that an a priori knowledge of the target program's behavior would increase the power of these mechanisms [2,3], no further investigations have been pursued in order to take full advantage of this idea in the context of runtime monitoring.

In this paper, we present an approach to generate a safe instrumented program, from a security policy and an untrusted program in which the monitor draws on an a priori knowledge of the program's possible behavior. The policy is stated as a deterministic Rabin automaton, a model which can recognize the same class of languages as non deterministic Büchi automata [4].

A. Jøsang, T. Maseng, and S.J. Knapskog (Eds.): NordSec 2009, LNCS 5838, pp. 287–301, 2009.

In our framework a program execution may be of infinite length representing the executions of programs such as daemons or servers. Finite executions are made infinite by attaching at their end an infinite repetition of a void action.

The use of Rabin automaton is motivated by the need for determinism in order to simplify our method and the associated proofs.

Our approach draws on advances in discrete events system control by [5] and on related subsequent research by Langar and Mejri [6] and consists in combining two models via the automata product operator: a model representing the system's behavior and another one representing the property to be enforced. In our approach, the model representing the system's behavior is represented by a LTS and the property to be enforced is stated as a Rabin automaton. The LTS representing the program could be built directly from the control flow graph after a control flow analysis [7,8].

To sum up, our approach either returns an instrumented program, modeled as a labeled transition system, which provably respects the input security policy or terminates with an error message. While the latter case sometimes occurs, it is important to stress that this will never occur if the desired property is a safety property which can be enforced using existing approaches. Our approach is thus strictly more expressive.

The rest of this paper is organized as follows. Section 2 presents a review of related work. In Section 3, we define some concepts that are used throughout the paper. The elaborated method is presented in Section 4. In Section 5, we discuss the theoretical underpinnings of our method. Some concluding remarks are finally drawn in Section 6 together with an outline of possible future work.

2 Related Work

Schneider, in his seminal work [1], was the first to investigate the question of which security policies could be enforced by monitors. He focused on specific classes of monitors, which observe the execution of a target program with no knowledge of its possible future behavior and with no ability to affect it, except by aborting the execution. Under these conditions, he found that a monitor could enforce the precise security policies that are identified in the literature as safety properties, and are informally characterized by prohibiting a certain bad thing from occurring in a given execution. These properties can be modeled by a security automaton and their representation has formed the basis of several practical as well as theoretical monitoring frameworks.

Schneider's study also suggested that the set of properties enforceable by monitors could be extended under certain conditions. Building on this insight, Ligatti, Bauer and Walker [3,9] examined the way the set of policies enforceable by monitors would be extended if the monitor had some knowledge of its target's possible behavior or if its ability to alter that behavior were increased. The authors modified the above definition of a monitor along three axes, namely (1) the means on which the monitor relies in order to respond to a possible violation of the security policy; (2) whether the monitor has access to information about the program's possible behavior; (3) and how strictly the monitor is required to enforce the security policy. Consequently, they were able to provide a rich taxonomy of classes of security policies, associated with the appropriate

model needed to enforce them. Several of these models are strictly more powerful than the security automata developed by Schneider and are used in practice.

Evolving along this line of inquiry, Ligatti et al. [10] gave a more precise definition of the set of properties enforceable by the most powerful monitors, while Fong [11] and Talhi et al. [12] expounded on the capabilities of monitors operating under memory constraints. Hamlen et al. [2] , on the other hand showed that in-lined monitors, (whose operation is injected into the target program's code, rather than working in parallel), can also enforce more properties than those modeled by a security automaton. In [13], a method is given to enforce both safety and co-safety properties by monitoring.

The first practical application using this framework was developed by Erlingsson and Schneider in [14]. In that project, a security automaton is merged into object code, and static analysis is used to reduce the runtime overhead incurred by the policy enforcement. Similar approaches, working on source code, were developed by Colcombet and Fradet [15], by Langar and Mejri [6] and by Kim et al. [16,17,18,19]. All these methods are limited to enforcing safety properties, which must be included either as a security automaton, or stated in a custom logic developed for this application. The first two focus on optimizing the instrumentation introduced in the code.

3 Preliminaries

Before moving on, let us briefly start with some preliminary definitions.

We express the desired security property as a Rabin automaton. A Rabin automaton $\mathcal{R}$, over alphabet $\mathcal{A}$ is a tuple (Q, q_0, δ, C) such that

- $\mathcal{A}$ is a finite or countably infinite set of symbols;
- Q is a finite set of states;
- $q_0 \in Q$ is the initial state;
- $\delta \subseteq Q \times A \times Q$ is a transition function;
- $C = \{(L_j, U_j) | j \in J\}$ is the acceptance set. It is a set of couples (L_j, U_j) where $L_j \subseteq Q$ and $U_j \subseteq Q$ for all $j \in J$ and $J \subseteq \mathbb{N}$.

Let $\mathcal{R}$ stand for a Rabin automaton defined over alphabet $\mathcal{A}$. A subset $Q' \subseteq Q$ is *admissible* if and only if there exists a $j \in J$ such that $Q' \cap L_j = \emptyset$ and $Q' \cap U_j \neq \emptyset$.

For the sake of simplicity, we refer to the elements defining an automaton or a model following formalism: the set of states Q of automaton $\mathcal{R}$ is referred to as $\mathcal{R}.Q$ and we leave it as Q when it is clear in the context.

A *path* π, is a finite (respectively infinite) sequence of states $\langle q_1, q_2, ..., q_n \rangle$ (respectively $\langle q_1, q_2, ... \rangle$) such that there exists a finite (respectively infinite) sequence of symbols $a_1, a_2, ..., a_n$ (respectively $a_1, a_2, ...$) called the label of π such that $\delta(q_i, a_i) = q_{i+1}$ for all $i \in \{0, ..., n\}$ (respectively $i \geq 0$). In fact, a path is a sequence of states consisting of a possible run of the automaton, and the label of this path is the input sequence that generates this run. A path is said to be empty if its label is the empty sequence ϵ.

We denote by $set(\pi)$ the set of states visited by the path π. The first state of π is called the origin of π. If π is finite, the last state it visits is called its end; otherwise, if it is infinite, we write $inf(\pi)$ for the set of states that are visited infinitely often in π. A

path π is *initial* if and only if its origin is q_0, the initial state of the automaton, and it is *final* if and only if it is infinite and $inf(\pi)$ is admissible.

A path is *successful* if and only if it is both initial and final. The property of successfulness of a path determines, in fact, the acceptance condition of Rabin automata. A sequence is accepted by a Rabin automaton *iff* it is the label of a *successful* path.

The set of all accepted sequences of $\mathcal{R}$ is the language recognized by $\mathcal{R}$, noted $\mathcal{L}_{\mathcal{R}}$.

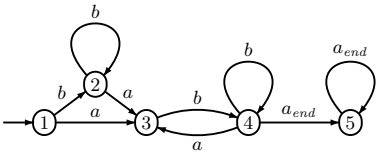
Let $q \in Q$ be a state of $\mathcal{R}$. We say that q is accessible *iff* there exists an initial path (possibly the empty path) that visits q. We say that q is co-accessible *iff* it is the origin of a final path.

Executions are modeled as sequences of atomic actions taken from a finite or countably infinite set of actions $\mathcal{A}$. The empty sequence is noted ϵ, the set of all finite length sequences is noted $\mathcal{A}^*$, that of all infinite length sequences is noted $\mathcal{A}^\omega$, and the set of all possible sequences is noted $\mathcal{A}^\infty = \mathcal{A}^\omega \cup \mathcal{A}^*$. Let $\tau \in \mathcal{A}^*$ and $\sigma \in \mathcal{A}^\infty$ be two sequences of actions. We write $\tau;\sigma$ for the concatenation of τ and σ. We say that τ is a prefix of σ noted $\tau \preceq \sigma$ *iff* $\tau \in \mathcal{A}^*$ and there exists a sequence σ' such that $\tau;\sigma' = \sigma$.

Let $a \in \mathcal{A}$ be an action symbol. A state $q' \in Q$ is an $a-$successor of q if $\delta(q,a) = q'$. Conversely, a state q' is a successor of q if there exists a symbol a such that $\delta(q,a) = q'$.

Let $\pi = \langle q_1, q_2, ..., q_n \rangle$ be a finite path in $\mathcal{R}$. This path is a cycle if $q_1 = q_n$.

The cycle π is admissible *iff* $set(\pi)$ is admissible. It is accessible *iff* there is a state q in $set(\pi)$ such that q is accessible, and likewise, it is co-accessible *iff* there is a state q in $set(\pi)$ such that q is co-accessible.



$C = \{(\{3\}, \{4\}), (\emptyset, \{5\})\}$

Fig. 1. A Rabin Automaton with acceptance Condition C

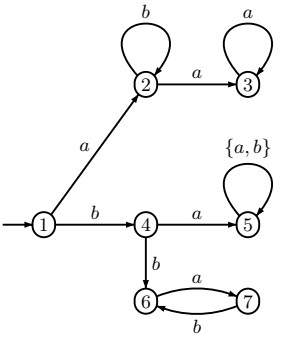


Fig. 2. Example- Labeled transition system

Let us consider Figure 1. It represents a Rabin automaton. In this figure, all the states are accessible and co-accessible. The paths $\langle 3,4,3,4,3 \rangle$, $\langle 3,4,3 \rangle$ and $\langle 2,2 \rangle$ are inadmissible cycles, while $\langle 5,5 \rangle$ is an admissible cycle and both infinite paths $\langle 1,2,3,4,5,5,\ldots \rangle$ and $\langle 1,2,3,4,3,4,4,\ldots \rangle$ are initial and final and therefore both are successful.

Finally a security property $\hat{\mathcal{P}}$ is a predicate on executions. An execution σ is said to be valid or to respect the property if $\hat{\mathcal{P}}(\sigma)$. A Rabin automaton $\mathcal{R}$ represents a security

policy $\hat{\mathcal{P}}$ *iff* $\mathcal{L}_{\mathcal{R}} = \{\sigma | \hat{\mathcal{P}}(\sigma)\}$, the set of executions that satisfy the security policy. Abusing the notation, we extend the application of $\hat{\mathcal{P}}$ to a set of sequences, thus if Σ is a set of sequences $\hat{\mathcal{P}}(\Sigma)$ means that all the sequences of Σ satisfy $\hat{\mathcal{P}}$.

4 Method

In this section we explain our approach in more detail and illustrate its operation with an example. The main algorithm takes as input a Rabin automaton $\mathcal{R}$, which represents a security Policy $\hat{\mathcal{P}}$ and a labeled transition system (LTS) $\mathcal{M}$, which models a program. The algorithm either returns a model of an instrumented program that enforces $\hat{\mathcal{P}}$ on $\mathcal{M}$ or returns an error message. The latter case occurs when it is not possible to produce an instrumented program that both enforces the desired security property and generates all valid sequences of $\mathcal{M}$.

Following [20,2,9], we consider that an enforcement mechanism successfully, enforces the property if the two following conditions are satisfied. First, the enforcement mechanism must be transparent; meaning that all possible program executions that respect the property must be emitted, i.e. the enforcement mechanism cannot prevent the execution of a sequence satisfying the property. Second, the enforcement mechanism must be sound, meaning that it must ensure that all observable output respects the property. We revisit and expand these ideas in Sections 4.3 and 5. We illustrate each step of our approach using an example program and a security policy.

4.1 Property Encoding

As mentioned earlier, the desired security property is stated as a Rabin automaton. The security property $\hat{\mathcal{P}}$ to which we seek to conform the target program is modeled by the Rabin automaton in Figure 1, over the alphabet $\mathcal{A} \cup \{a_{end}\}$ with $\mathcal{A} = \{a, b\}$. The symbol a_{end} is a special token added to $\mathcal{A}$ to capture the end of a finite sequence, since the Rabin automaton only accepts infinite length sequences. The finite sequence σ is thus modeled as $\sigma; (a_{end})^{\omega}$. The language accepted by this automaton is the set of executions that containing only a finite non-empty number of a actions and such that finite executions end with a b action.

For the sake of simplicity, if a sequence $\sigma = \tau; (a_{end})^{\omega}$ with $\tau \in \mathcal{A}^*$ is such that $\hat{\mathcal{P}}(\sigma)$ we say that $\hat{\mathcal{P}}(\tau)$.

4.2 Program Abstraction

The program is abstracted as a labeled transition system (LTS). This is a conservative abstraction, widely used in model checking and static analysis, in which a program is abstracted as a graph, whose nodes represent program points, and whose edges are labeled with instructions (or abstractions of instructions, or actions). Formally, a LTS $\mathcal{M}$, over alphabet $\mathcal{A}$ is a deterministic graph (Q, q_0, δ) such that:

- $\mathcal{A}$ is a finite or countably infinite set of actions;
- Q is a finite set of states;

- q_0 is the initial state;
- $\delta : Q \times \mathcal{A} \times Q$ is a transition function. For each $q \in Q$, there must be at least one $a \in \mathcal{A}$ for which $\delta(q, a)$ is defined.

Here also a finite sequence σ is extended with the suffix $(a_{end})^\omega$ yielding the infinite sequence $\sigma; (a_{end})^\omega$.

In general, static analysis tools do not always generate deterministic LTSs. Yet, this restriction can be imposed with no loss of generality. Indeed, a non-deterministic LTS $\mathcal{M}$ over alphabet $\mathcal{A}$ can be represented by an equivalent deterministic LTS $\mathcal{M}'$ over alphabet $\mathcal{A} \times \mathbb{N}$, which is equivalent to $\mathcal{M}$ if we ignore the numbers $i \in \mathbb{N}$ associated with the actions. Each occurrence of an action a is associated with a unique index in $\mathbb{N}$ so as to distinguish it from other occurrences of the same action a. In what follows, we can thus consider only deterministic LTSs. Furthermore, we focus exclusively on infinite length executions.

The example program that we use to illustrate our approach is modeled by the LTS in Figure 2, over the alphabet $\mathcal{A}$. The issue consisting of how to abstract a program into a LTS is beyond the scope of this paper.

As with the Rabin Automata, we define a *path* π as a finite or infinite sequence of states $\langle q_1, q_2, \ldots \rangle$ such that there exists a corresponding sequence of actions $(a_1, a_2 \ldots)$ called the label of π, for which the $\delta(q_i, a_i) = q_{i+1}$.

The set of all labels of infinite paths starting in q_0 is the language generated or emitted by $\mathcal{M}$ and is noted $\mathcal{L}_\mathcal{M}$.

4.3 Algorithm

The algorithm's input consists of the program model $\mathcal{M}$ and a Rabin automaton $\mathcal{R}$ which encodes the property. The output is a truncation automaton $\mathcal{T}$ representing a model of an in-lined monitored program acting exactly identically to the input program for all the executions satisfying the property and halting a bad execution after producing a valid prefix of this execution.

A high level description of the algorithm is as follows:

1. Build a product automaton $\mathcal{R}^P$ whose recognized language is exactly : $\mathcal{L}_{\mathcal{R}^P} = \mathcal{L}_\mathcal{R} \cap \mathcal{L}_\mathcal{M}$.
2. Build $\mathcal{R}^T$ from $\mathcal{R}^P$ by the application of a transformation allowing it to accept partial executions of the program modeled by $\mathcal{M}$ that satisfy the property $\hat{\mathcal{P}}$.
3. Check if $\mathcal{R}^T$ could be used as a truncation automaton and produce a LTS $\mathcal{T}$ modeling the program instrumented by a truncation mechanism otherwise produce **error**.

The following sections give more details on each step.

Automata Product. The first phase of the transformation is to construct $\mathcal{R}^P$, a Rabin automaton that accepts the intersection of the language accepted by the automaton $\mathcal{R}$ and the language emitted by $\mathcal{M}$. This is exactly the product of these two automata. Thus $\mathcal{R}^P$ accepts the set of executions that both respect the property and represent executions of the target program.

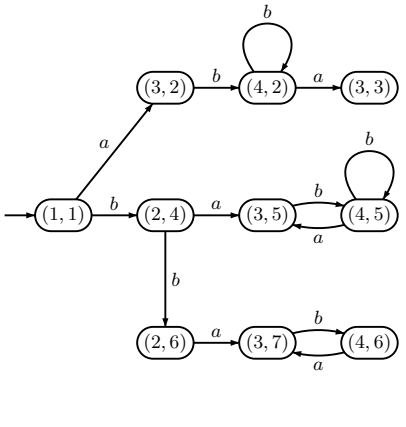
Given a property automaton $\mathcal{R} = (\mathcal{R}.Q, \mathcal{R}.q_0, \mathcal{R}.\delta, \mathcal{R}.C)$ and a Labeled Transition system $\mathcal{M} = (\mathcal{M}.Q, \mathcal{M}.q_0, \mathcal{M}.\delta)$ the automaton $\mathcal{R}^P$ is constructed as follows:

- $\mathcal{R}^P.Q = \mathcal{R}.Q \times \mathcal{M}.Q$
- $\mathcal{R}^P.q_0 = (\mathcal{R}.q_0, \mathcal{M}.q_0)$
- $\forall q \in R.Q, q' \in \mathcal{M}.Q \wedge a \in (A \cup \{a_{end}\})$

$$\mathcal{R}^P.\delta((q,q'),a) = \begin{cases} (\mathcal{R}.\delta(q,a), \mathcal{M}.\delta(q',a)) & \text{if } \mathcal{R}.\delta(q,a) \text{ and } \mathcal{M}.\delta(q',a) \text{ are defined} \\ \text{undefined otherwise} \end{cases}$$

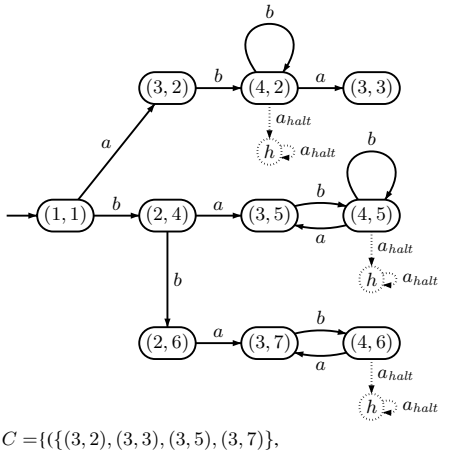
- $\mathcal{R}^P.C = \bigcup_{(L,U)\in\mathcal{R}.C}\{(L \times \mathcal{M}.Q, U \times \mathcal{M}.Q)\}$

The automaton built for our example using the property in Figure 1 and the program model presented in Figure 2 is given in Figure 3.



$C = \{(\ \{(3,2),(3,3),(3,5),(3,7)\},$
$\{(4,2),(4,5),(4,6)\}\ )\}$

Fig. 3. Example - Rabin automaton $\mathcal{R}^P$



$C = \{(\{(3,2),(3,3),(3,5),(3,7)\},$
$\{(4,2),(4,5),(4,6)\}),(\emptyset,\{h\})\}$

Fig. 4. Transformed Product Automaton

Since $\mathcal{R}^P$ accepts the intersection of the languages accepted by the automaton $\mathcal{R}$ and $\mathcal{M}$, it would seem an ideal abstraction from which to build the instrumented program. However, there is no known way to transform such an automaton into a program. Indeed, since the acceptance condition of the Rabin automaton is built around the notion of infinite traces reaching some states infinitely often, a dynamic monitoring system built from such an automaton with no help provided by a prior static analysis, may never be able to determine if a given execution is valid or not.

Instead, we extract a deterministic automaton, $\mathcal{T} = (\mathcal{T}.Q, \mathcal{T}.q_0, \mathcal{T}.\delta)$, from the Rabin automaton $\mathcal{R}^P$. This automaton is the labeled transition system which is returned. It forms in turn the basis of the instrumented program we seek to construct. The instrumented program is expected to work as a program monitored by a truncation automaton meaning that its model $\mathcal{T}$ has to satisfy the following conditions: (1) $\mathcal{T}$ emits each

execution of $\mathcal{M}$ satisfying the security property without any modification, (2) for each execution that does not satisfy the property, $\mathcal{T}$ safely halts it after producing a valid partial execution, and (3) $\mathcal{T}$ does not emit anything else apart those executions described in (1) and (2).

The next step toward this goal is to apply a transformation that allows $\mathcal{R}^P$ to accept partial executions of $\mathcal{M}$ which satisfy the property. Indeed, all finite initial paths in $\mathcal{R}^P$ represent partial executions of $\mathcal{M}$, only some of them satisfy the security property. We add a transition, labeled a_{halt}, to a new state h to every state in R^P where the execution could be aborted after producing a partial execution satisfying the property, i.e. a state (q_1, q_2) for which $\mathcal{R}.\delta(q_1, a_{end})$ is defined. The state h is made admissible by adding the transition (h, a_{halt}, h) to the set of transitions and the pair $(\emptyset, \{h\})$ to the acceptance set. We have to be careful in choosing h and a_{halt} such that $h \notin \mathcal{R}.Q \cup \mathcal{M}.Q$ and $a_{halt} \notin \mathcal{A}$ the alphabet of actions.

We refer to this updated version of $\mathcal{R}^P$ as $\mathcal{R}^T$, built from $\mathcal{R}^P$ as follows :

- $\mathcal{R}^T.Q = \mathcal{R}^P.Q \cup \{h\}$
- $\mathcal{R}^T.q_0 = \mathcal{R}^P.q_0$
- $\mathcal{R}^T.\delta = \mathcal{R}^P.\delta \cup \{(q, a_{halt}, h) | \mathcal{R}^P.\delta(q, a_{end})$ is defined $\} \cup \{(h, a_{halt}, h)\}$.
- $\mathcal{R}^T.C = \mathcal{R}^P.C \cup \{(\emptyset, \{h\})\}$

After this transformation our example product automaton becomes the automaton depicted in Figure 4. The halt state h has been duplicated three times in order to avoid cross edging.

The language recognized by $\mathcal{R}^T$ is

$\mathcal{L}_{\mathcal{R}^T} = (\mathcal{L}_{\mathcal{R}} \cap \mathcal{L}_{\mathcal{M}}) \cup \{\tau; (a_{halt})^\omega | (\tau \in \mathcal{A}^*) \wedge (\exists \sigma \in \mathcal{L}_{\mathcal{M}} : \tau \preceq \sigma) \wedge (\tau; (a_{end})^\omega \in \mathcal{L}_{\mathcal{R}})\}$.

Extracting a Model of the Instrumented Program. The next phase consists in extracting, if possible, a labeled transition system $\mathcal{T} = (Q, q_0, \delta)$, from the Rabin automaton $\mathcal{R}^T$. This automaton is expected to behave as the original program monitored by a truncation automaton.

To understand the need for this step, first note that the acceptance condition of a Rabin automaton could not be checked dynamically due to its infinite nature. Should we build an instrumented program directly from $\mathcal{R}^T$, by ignoring its acceptance condition, and treating it like a simple LTS, the resulting program would still generate all traces of $\mathcal{M}$ that verify the property $\hat{\mathcal{P}}$ but it would also generate the invalid sequences of $\mathcal{M}$ representing labels of infinite paths in $\mathcal{R}^T$ trapped in non admissible cycles. In other words, the enforcement of the property would be transparent but not sound.

In order to generate $\mathcal{T}$, we prune $\mathcal{R}^T$ of some of its states and transitions, eliminating inadmissible cycles while taking care to preserve the ability to generate all the valid sequences of $\mathcal{L}_{\mathcal{M}}$. Furthermore, we need to ascertain that $\mathcal{T}$ aborts the execution of every sequence of $\mathcal{L}_{\mathcal{M}}$ not satisfying $\hat{\mathcal{P}}$ and that $\mathcal{T}$ generates only executions satisfying $\hat{\mathcal{P}}$.

We can now restate the correctness requirements of our approach. In the formulation of these requirements, the actions a_{end} and a_{halt} are ignored, as they merely model the end of a finite sequence.

$$(\forall \sigma \in \mathcal{L}_{\mathcal{M}}| : (\exists \tau \in \mathcal{L}_{\mathcal{T}}| : ((\tau = \sigma) \vee (\tau \preccurlyeq \sigma)) \wedge \hat{\mathcal{P}}(\tau) \wedge (\hat{\mathcal{P}}(\sigma) \Longrightarrow (\tau = \sigma)))) \tag{4.1}$$

$$\forall \tau \in \mathcal{L}_{\mathcal{T}}| : ((\exists \sigma \in \mathcal{L}_{\mathcal{M}}| : ((\tau = \sigma) \vee (\tau \preccurlyeq \sigma))) \wedge \hat{\mathcal{P}}(\tau) \tag{4.2}$$

Note that the requirements 4.1 and 4.2 are not only sufficient to ensure the respect of soundness and transparency requirements introduced at the beginning of Section 4 following [20,2,9], but also that of a more restrictive requirement. Indeed, requirement 4.1 also states that the mechanism is a truncation mechanism. It ensures the compliance to the security property by aborting the execution before a security violation occurs whenever this is needed. We can thus prove that for any invalid sequence present in the original model, the instrumented program outputs a valid prefix of that sequence.

Our enforcement mechanism is not allowed to generate sequences that are not related to sequences in $\mathcal{L}_{\mathcal{M}}$ either by equality or prefix relation. Furthermore these sequences must satisfy $\hat{\mathcal{P}}$. This is stated in requirement 4.2.

Requirements 4.1 and 4.2 give the guidelines for constructing $\mathcal{T}$ from $\mathcal{R}^T$. The transformations that are performed on $\mathcal{R}^T$ to ensure meeting these requirements are elaborated around the following intuition. The automaton $\mathcal{R}^T$ has to be pruned so as to ensure that it represents a safety property even though $\mathcal{R}$ is not. Note that this is not possible in the general case without violating the requirements. The idea is that admissible cycles are visited infinitely often by executions satisfying $\hat{\mathcal{P}}$ and must thus be included in $\mathcal{T}$. Likewise, any other state or transition that can reach an admissible cycle may be part of such an execution and must be included. On the other hand, inadmissible cycles cannot be included in $\mathcal{T}$ as the property is violated by any trace that goes through such a cycle infinitely often. In some cases their elimination cannot occur without the loss of transparency and our approach fails, returning **error**. The underlying idea of the subsequent manipulation is thus to check whether we can trim $\mathcal{R}^T$ by removing bad cycles but without also removing the states and transitions required to ensure transparency.

The following steps show how we perform the trim procedure.

The nest step is to determine the strongly connected components (*scc*) in the graph representing $\mathcal{R}^T$ using Tarjan's algorithm [21]. We then examine each *scc* and mark it as containing either only admissible cycles, only inadmissible cycles, both types of cycles, or no cycles (in the trivial case).To perform this last operation, we have developed heuristics based on the notion that graphs which model programs are structured. A discussion of these heuristics is however beyond the scope of this paper.

The next step is to construct the quotient graph of $\mathcal{R}^T$ in which each node represents a *scc* and an edge connecting two *scc* c_1 and c_2 indicates that there exists a state q_1 in *scc* c_1 and a state q_2 in *scc* c_2 and an action a such that $\mathcal{R}^T.\delta(q_1, a) = q_2$. We assume, without loss of generality, that all the *scc* states are accessible from the initial node, the *scc* containing q_0.

The nodes of the quotient graph $\mathcal{R}^T$ are then visited in reverse topological ordering. We determine for each one whether it should be kept intact, altered or removed.

In the what follows the *scc* containing the halting state h is referred to as H.

A *scc* with no cycle at all is removed with its incident edges if it cannot reach another *scc*. In Figure 4 the *scc* consisting of the state $(3, 3)$ would thus be eliminated.

A *scc* containing only admissible cycles should be kept, since all the executions reaching it satisfy $\hat{\mathcal{P}}$. Eliminating it would prevent the enforcement mechanism from

being transparent. In our example in Figure 4 the *scc* consisting of the single state $(4, 2)$ has only admissible cycles and should be kept.

A *scc* containing only non admissible cycles can be removed if it cannot reach another *scc* with only admissible cycles. Otherwise, we are generally forced to return **error**. However, in some cases, we can either break the inadmissible cycles or prevent them from reaching H by removing some transitions and keeping the remainder of the *scc*. This occurs when the only successor, having admissible cycles, of this *scc* is H. In our example, the *scc* containing the states $(3, 7)$ and $(4, 6)$ has only non admissible cycles and H is its only successor. We can eliminate this *scc* and halt with **error** at this point. Yet, if we observe that eliminating the transition $((4, 6), a, (3, 7))$ would break the inadmissible cycle, we can eliminate that transition and keep the rest of the *scc*.

A transition can only be removed if its origin has h as immediate successor. This is because, should the instrumented program attempt to perform the action that corresponds to this transition, its execution would be aborted. However, a partial execution only satisfies the property if it ends in a state that has h as an immediate successor.

A *scc* containing admissible and non admissible cycles may cause good or bad behavior. Actually, an execution reaching this *scc* may be trapped in an inadmissible cycle for ever or may leave it to reach an admissible cycle thus satisfying the property $\hat{\mathcal{P}}$. We have no means to dynamically check whether the execution is going to leave a cycle or not. Thus, in this case we must abort with **error**. In the example given in Figure 4 the *scc* consisting of the two states $(3, 5)$ and $(4, 5)$ have one admissible cycle, $\langle(4, 5), (4, 5)\rangle$ and one inadmissible cycle $\langle(3, 5), (4, 5), (3, 5)\rangle$. This last cycle is visited if the invalid sequence $(ba)^\omega$ is being generated. Note that the automaton accepts an infinite number of valid traces of the form $ba(ba)^*b^\omega$, and that no truncation automaton can both accept these traces and reject the invalid trace described above. Hence we have to abort the algorithm with **error** in such cases.

After removing all the *scc* with inadmissible cycles and provided we have not aborted, we can be sure that an instrumented program built from $\mathcal{T}$ would not contain any infinite length execution which does not respect the security property. We must still verify that whenever the execution is halted, the partial sequence emitted satisfies $\hat{\mathcal{P}}$.

The last step is to check whether the eliminated states and transitions could not allow invalid partial executions to be emitted. This verification is based on the following observation: if a removed transition has an origin state that is not an immediate predecessor of h this would then allow to emit a partial execution that does not satisfy $\hat{\mathcal{P}}$. Hence, the verification merely consists in checking whether we have removed transitions from states that are not immediate predecessors of h; if such is the case we have to abort with **error**. More precisely, for a state $q = (q_1, q_2)$ in $\mathcal{T}$ we have to check whether it is possible from q_2 in $\mathcal{M}$ to perform actions that are not possible from q; if this is the case, q must have h as immediate successor; otherwise, we have no other option than to terminate the algorithm without returning a suitable LTS and with an error message.

We may also remove the transitions of the form (h, a_{halt}, h) and (q, a_{end}, q), where $q \in \mathcal{R}^T.Q$.

5 Mechanism's Enforcement Power

In this section, we show that non-uniform enforcement mechanisms, which occur when the set of possible executions Σ is a subset if A^ω, are more powerful than uniform enforcers, i.e. those for which $\Sigma = A^\omega$, in the sense that they are able to enforce a larger class of security properties. This demonstration will reveal that monitors that are tailored to specific programs may be able to enforce a wide set of properties and argues for the use of static analysis in conjunction with monitoring.

Let us begin with a more formal definition of the concepts we discussed in the previous sections, following the notations adopted in [3,9]. We specify the enforcement mechanism behavior of a security automaton S by judgments of the form $(q, \sigma) \xrightarrow{\tau}_S (q', \sigma')$ where q is the current state of the automaton; σ is the attempted execution; q' is the state the automaton reach after one execution step; σ' is the remaining execution trace to be performed; and τ is the execution trace consisting of one action at most that is emitted by the security automaton after one step.

The execution of the security automaton is generalized with the multi-step judgments defined through reflexivity and transitivity rules as follows.

Definition 1 (Multi-step semantics). *Let S be a security automaton. The multi-step relation $(q, \sigma) \overset{\tau}{\Longrightarrow}_S (q', \sigma')$ is inductively defined as follows.*
For all $q, q', q'' \in Q$, $\sigma, \sigma', \sigma'' \in \mathcal{A}^\infty$ and $\tau, \tau' \in \mathcal{A}^*$ we have

$$(q, \sigma) \overset{\varepsilon}{\Longrightarrow}_S (q, \sigma) \tag{5.1}$$

$$\text{if } (q, \sigma) \overset{\tau}{\Longrightarrow}_S (q'', \sigma'') \text{ and } (q'', \sigma'') \xrightarrow{\tau'}_S (q', \sigma') \text{ then } (q, \sigma) \overset{\tau;\tau'}{\Longrightarrow}_S (q', \sigma') \tag{5.2}$$

We are now able to give the definition of what a security enforcement mechanism is. Intuitively, we can think of security enforcement mechanisms as sequence transformers, automata that take a program's actions sequence as input, and output a new sequence of actions that respects the security property. This intuition is formalized as follows:

Definition 2 (Transformation). *A security automaton $S = (Q, q_0, \delta)$ transforms an execution trace $\sigma \in \mathcal{A}^\infty$ into an execution $\tau \in \mathcal{A}^\infty$, noted $(q_0, \sigma) \Downarrow_S \tau$, if and only if*

$$\forall q \in Q, \sigma' \in \mathcal{A}^\infty, \tau' \in \mathcal{A}^* \mid: ((q_0, \sigma) \overset{\tau'}{\Longrightarrow}_S (q', \sigma')) \Longrightarrow \tau' \preccurlyeq \tau \tag{5.3}$$

$$\forall \tau' \preccurlyeq \tau \mid: \exists q' \in Q, \sigma' \in \mathcal{A}^\infty \mid: (q_0, \sigma) \overset{\tau'}{\Longrightarrow}_S (q', \sigma') \tag{5.4}$$

We have seen that a security enforcement mechanism must respect two properties namely soundness and transparency. The former requires that no invalid execution be permitted, while the latter states that all valid executions must be transformed into semantically equivalent executions. But for enforcement to be meaningful, the notion of equivalence must be constrained. Otherwise, one might argue, for instance, that the empty sequence ϵ is equivalent to every valid execution, and enforce *any* property by aborting every execution at its onset.

Instead, we argue that two executions $\tau, \sigma \in \mathcal{A}^\infty$ are equivalent if there exists a reflexive, symmetric and transitive, equivalence relation $\cong$ s.t. $\tau \cong \sigma$.

We can now state formally what it means for an enforcement mechanism to effectively enforce a security property

Definition 3 (effective$^{\Sigma}_{\cong}$Enforcement). *Let $\Sigma \subseteq \mathcal{A}^\infty$ be a set of execution traces. A security automaton $S = (Q, q_0, \delta)$* enforces effectively$_\cong$ *a security property $\hat{\mathcal{P}}$ for Σ if and only if for all input trace $\sigma \in \Sigma$ there exists an output trace $\tau \in \mathcal{A}^\infty$ such that*

$$(q_0, \sigma) \Downarrow_S \tau \tag{5.5}$$

$$\hat{\mathcal{P}}(\tau) \tag{5.6}$$

$$\hat{\mathcal{P}}(\sigma) \Longrightarrow \sigma \cong \tau \tag{5.7}$$

Informally, a security automaton *enforces effectively*$_\cong$ a property for Σ *iff* for each execution trace $\sigma \in \Sigma$, it outputs a trace τ such that τ is valid, with respect to the property, and if the input trace σ is itself valid then $\sigma \cong \tau$.

Definition 4 ($\mathcal{S}^{\Sigma}_{\cong}$-enforceable). *Let $\Sigma \subseteq \mathcal{A}^\infty$ be a set of execution traces and $\mathcal{S}$ be a class of security automata. The class $\mathcal{S}^{\Sigma}_{\cong}$-enforceable is the set of security properties such that for each property in this set, there exists a security automaton $S \in \mathcal{S}$ that effectively$\cong$ enforces this property for the traces in Σ.*

Our approach is built around the idea, first suggested by Ligatti et al. in [3,9], that the set of properties enforceable by a monitor could sometimes be extended if the monitor has some knowledge of the program's possible behavior and thus can rule out some executions as impossible.

We can now state this idea more formally.

Theorem 1. *Let $\mathcal{S}$ be a class of security automata and let $\Sigma^\natural, \Sigma^\sharp \subseteq \mathcal{A}^\infty$ be two sets of execution traces $\Sigma^\natural \subseteq \Sigma^\sharp$ then we have*

$$\mathcal{S}^{\Sigma^\sharp}_{\cong}\text{-enforceable} \subseteq \mathcal{S}^{\Sigma^\natural}_{\cong}\text{-enforceable} \tag{5.8}$$

The proof is quite straightforward, and based upon the intuition that a security mechanism possessing certain knowledge about its target may discard it, and then behave as an enforcement mechanisms lacking this knowledge.The proof has been omitted for space consideration.

Corollary 1. *Let $\mathcal{S}$ be a class of security automaton. For all execution trace set $\Sigma \subseteq \mathcal{A}^\infty$ we have*

$$\mathcal{S}^{\mathcal{A}^\infty}_{\cong}\text{-enforceable} \subseteq \mathcal{S}^{\Sigma}_{\cong}\text{-enforceable} \tag{5.9}$$

Corollary 1 indicates that any security property that is effectively$_\cong$ enforceable by a security automaton in a uniform context ($\Sigma = \mathcal{A}^\infty$) is also enforceable in the nonuniform context ($\Sigma \neq \mathcal{A}^\infty$). It follows that our approach is at least as powerful as those previously suggested in the literature that we built around that last framework.

It would be interesting to prove that for all security automaton classes, $\mathcal{S}$ and for all equivalence relations $\cong$, we have $\mathcal{S}^{\mathcal{A}^\infty}_{\cong}$-enforceable $\subset \mathcal{S}^{\Sigma}_{\cong}$-enforceable.

This is unfortunately not the case, as there exists at least one class of security automaton (ex. $\mathcal{S} = \emptyset$), and one equivalence relation (ex. $\tau \cong \sigma \; \forall \tau, \sigma \in \mathcal{A}^\infty$) such that $\mathcal{S}^{\mathcal{A}^\infty}_{\cong}$-enforceable $= \mathcal{S}^{\Sigma}_{\cong}$-enforceable for all set of traces $\Sigma \subseteq \mathcal{A}^\infty$.

However in our approach, we focus both on a specific class of security automata and on a specific equivalence relation. In our particular case, the set of policies enforceable in a nonuniform context is strictly greater than the one that is enforceable in the uniform context.

The monitors used in this paper are truncation automata, first described in [1]. These are monitors which, when presented with a potentially invalid sequence, have no option but to abort the execution.

Definition 5 (Truncation Automaton). *A truncation automaton is a security automaton where* $\delta : Q \times \mathcal{A} \rightarrow Q \cup \{halt\}$ *and* $halt \notin Q$.

Furthermore, we use syntactic equivalence (=) as the equivalence relation between valid traces.

We can now state the central theorem of this paper, that the enforcement power of the truncation automaton is strictly greater in the nonuniform context than in the uniform context, when we consider =-enforcement.

Theorem 2. *For all set of traces* $\Sigma \subset \mathcal{A}^\infty$ *we have*

$$\mathsf{T}^{\mathcal{A}^\infty}_{=}\text{-enforceable} \subset \mathsf{T}^{\Sigma}_{=}\text{-enforceable} \tag{5.10}$$

The proof is based on the following observations. First, it has been shown in [1,3] that a property is $\mathsf{T}^{\mathcal{A}^\infty}_{=}$-enforceable *iff* it is a safety property. Second. Let $\hat{\mathcal{P}}$ be a security property, $\hat{\mathcal{P}}$ is trivially enforceable on Σ *iff* for every sequence $\sigma \in \Sigma$, $\hat{\mathcal{P}}(\sigma)$. The proof thus consists in showing that for any $\Sigma \subset \mathcal{A}^\infty$, a nonsafety property can be stated, and trivially enforced.More specifically, this proof seeks to demonstrate for a sequence $\upsilon \in \mathcal{A}^\infty$ s. t. $\upsilon \notin \Sigma$ the non-safety security property $\hat{\mathcal{P}}(\sigma) \iff (\sigma \neq \upsilon)$ for all $\sigma \in \mathcal{A}^\infty$ is $\mathsf{T}^{\Sigma}_{=}$-enforceable. The proof has been omitted for space consideration.

6 Conclusion and Future Work

The main contribution of this paper is the elaboration of a method aiming at in-lining a security enforcement mechanism in an untrusted program. The security property to be enforced is expressed by a Rabin automaton and the program is modeled by a LTS. The in-lined monitoring mechanism is actually a truncation mechanism allowing valid executions to run normally while halting bad executions before they violate the property.

In our approach, the monitor's enforcement power is extended by giving it access to statically gathered information about the program's possible behavior. This allows us to enforce non-safety properties for some programs. Nevertheless, several cases still exist where our approach fails to find a suitable instrumented code. These are cases where an execution may alternate between satisfying the property or not and could halt in an invalid state, or cases where an invalid execution contains no valid prefixes where the execution could be aborted without also ruling out some valid executions.

Another contribution of this study is to provide a proof that a truncation mechanism that effectively enforces a security property under the equality as an equivalence relation is strictly more powerful in a non uniform context than in a uniform one.

A more elaborate paradigm dealing with what constitutes a monitor could allow us to ensure the satisfaction of the security property in at least some cases where doing so in currently not feasible. For example, the monitor could suppress a sub-sequence of the program, and keep it under observation until it is satisfied that the program actually satisfies the property and output it all at once. Alternatively, the monitor may be allowed to insert some actions at the end of an invalid sequence in order to guarantee that the sequence is aborted in a valid state. Such monitors are suggested in [3], their use would extend this approach to a more powerful framework. Another question that remains open is to determine how often the algorithm will succeed in finding a suitable instrumented code when tested on real programs. We are currently developing an implementation to investigate this question further and hope to gain insights as to which of the above suggested extensions would provide the greatest increase in the set of enforceable properties.

Finally, a distinctive aspect of the method under consideration is that unlike other code instrumentation methods, ours induces no added runtime overhead. However, the size of the instrumented program is increased in the order $\mathcal{O}(m \times n)$, where m is the size of the original program and n is the size of the property. The instrumentation algorithm itself runs in time $\mathcal{O}(p \times c)$, where p is the size of the automaton's acceptance condition and c is the number of cycles in the product automaton. In practice, graphs that abstract programs have a comparatively small number of cycles.

References

1. Schneider, F.B.: Enforceable security policies. Information and System Security 3(1), 30–50 (2000)
2. Hamlen, K.W., Morrisett, G., Schneider, F.B.: Computability classes for enforcement mechanisms. ACM Transactions on Programming Languages and Systems (TOPLAS) 28(1), 175–205 (2006)
3. Bauer, L., Ligatti, J., Walker, D.: More enforceable security policies. In: Proceedings of the Foundations of Computer Security Workshop, Copenhagen, Denmark (July 2002)
4. Perrin, D., Pin, J.-É.: Infinite Words, ser. Pure and Applied Mathematics, vol. 141. Elsevier, Amsterdam (2004)
5. Ramadge, P.J., Wonham, W.M.: The control of discrete event systems. IEEE Proceedings: Special issue on Discrete Event Systems 77(1), 81–97 (1989)
6. Langar, M., Mejri, M.: Optimizing enforcement of security policies. In: Proceedings of the Foundations of Computer Security Workshop (FCS 2005) affiliated with LICS 2005 (Logics in Computer Science) (June-July 2005)
7. Aho, A.V., Sethi, R., Ullman, J.D.: Compilers, Principles, Techniques, and Tools. Addison-Wesley, Reading (1986)
8. Beyer, D., Henzinger, T.A., Jhala, R., Majumdar, R.: The software model checker BLAST: Applications to software engineering. International Journal on Software Tools for Technology Transfer (STTT) 9(5-6), 505–525 (2007)
9. Ligatti, J., Bauer, L., Walker, D.: Edit automata: Enforcement mechanisms for run-time security policies. International Journal of Information Security (2004)

10. Ligatti, J., Bauer, L., Walker, D.: Enforcing non-safety security policies with program monitors. In: di de Vimercati, S.C., Syverson, P.F., Gollmann, D. (eds.) ESORICS 2005. LNCS, vol. 3679, pp. 355–373. Springer, Heidelberg (2005)
11. Fong, P.: Access control by tracking shallow execution history. In: Proceedings of the 2004 IEEE Symposium on Security and Privacy, Oakland,California, USA (May 2004)
12. Talhi, C., Tawbi, N., Debbabi, M.: Execution monitoring enforcement under memory-limitations constraints. Information and Computation 206(1), 158–184 (2008)
13. Bauer, A., Leucker, M., Schallhart, C.: Monitoring of real-time properties. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 260–272. Springer, Heidelberg (2006)
14. Erlingsson, U., Schneider, F.B.: SASI enforcement of security policies: A retrospective. In: Proceedings of the WNSP: New Security Paradigms Workshop. ACM Press, New York (2000)
15. Colcombet, T., Fradet, P.: Enforcing trace properties by program transformation. In: Proceedings of the 27th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (January 2000)
16. Kim, M.: Information extraction for run-time formal analysis. Ph.D. dissertation, University of Pennsylvania (2001)
17. Kim, M., Viswanathan, M., Kannan, S., Lee, I., Sokolsky, O.: Java-mac: A run-time assurance approach for java programs. Formal Methods in Systems Design 24(2), 129–155 (2004)
18. Lee, I., Kannan, S., Kim, M., Sokolsky, O., Viswanathan, M.: Runtime assurance based on formal specifications. In: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (1999)
19. Sokolsky, O., Kannan, S., Kim, M., Lee, I., Viswanathan, M.: Steering of real-time systems based on monitoring and checking. In: Proceedings of the Fifth International Workshop on Object-Oriented Real-Time Dependable Systems, WORDS 1999, p. 11. IEEE Computer Society, Washington (1999)
20. Erlingsson, U.: The inlined reference monitor approach to security policy enforcement. Ph.D. dissertation, Cornell University, Ithaca, NY, USA (2004)
21. Tarjan, R.E.: Depth-first search and linear graph algorithms. SIAM Journal on Computing 1(2), 146–160 (1972)

Author Index