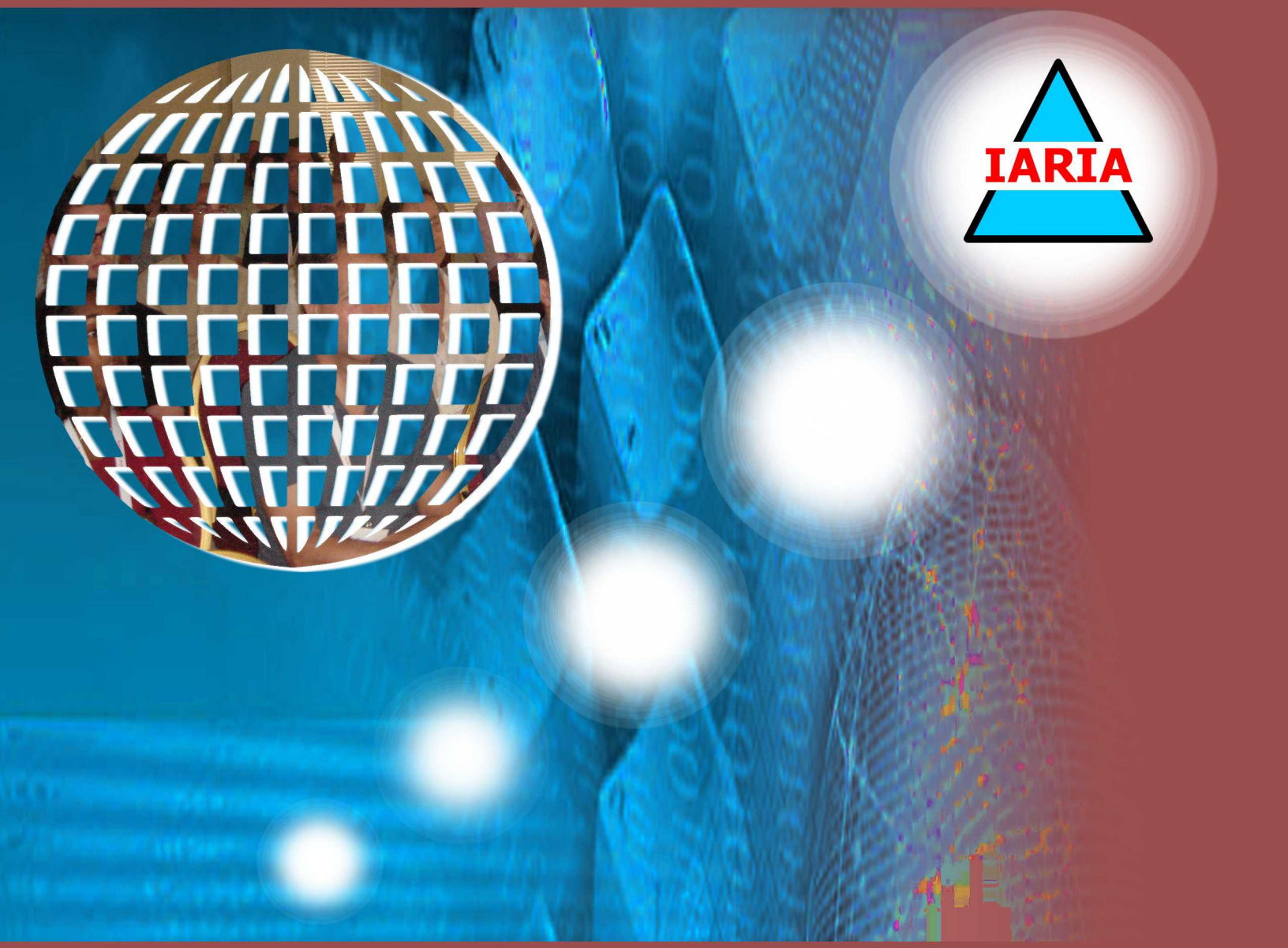
International Journal on Advances in Security



2015 vol. 8 nr. 1&2

The *International Journal on Advances in Security* is published by IARIA.
ISSN: 1942-2636
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Security
Volume 8, Number 1 & 2, 2015

Raimund K. Ege, Northern Illinois University, USA
Laila El Aimani, Technicolor, Security & Content Protection Labs., Germany
El-Sayed M. El-Alfy, King Fahd University of Petroleum and Minerals, Saudi Arabia
Rainer Falk, Siemens AG - Corporate Technology, Germany
Shao-Ming Fei, Capital Normal University, Beijing, China
Eduardo B. Fernandez, Florida Atlantic University, USA
Anders Fongen, Norwegian Defense Research Establishment, Norway
Somchart Fugkeaw, Thai Digital ID Co., Ltd., Thailand
Steven Furnell, University of Plymouth, UK
Clemente Galdi, Universita' di Napoli "Federico II", Italy
Emiliano Garcia-Palacios, ECIT Institute at Queens University Belfast - Belfast, UK
Birgit F. S. Gersbeck-Schierholz, Leibniz Universität Hannover, Certification Authority University of Hannover (UH-CA), Germany
Manuel Gil Pérez, University of Murcia, Spain
Karl M. Goeschka, Vienna University of Technology, Austria
Stefanos Gritzalis, University of the Aegean, Greece
Michael Grottke, University of Erlangen-Nuremberg, Germany
Ehud Gudes, Ben-Gurion University - Beer-Sheva, Israel
Indira R. Guzman, Trident University International, USA
Huong Ha, University of Newcastle, Singapore
Petr Hanáček, Brno University of Technology, Czech Republic
Gerhard Hancke, Royal Holloway / University of London, UK
Sami Harari, Institut des Sciences de l'Ingénieur de Toulon et du Var / Université du Sud Toulon Var, France
Dan Harkins, Aruba Networks, Inc., USA
Ragib Hasan, University of Alabama at Birmingham, USA
Masahito Hayashi, Nagoya University, Japan
Michael Hobbs, Deakin University, Australia
Neminath Hubballi, Infosys Labs Bangalore, India
Mariusz Jakubowski, Microsoft Research, USA
Ángel Jesús Varela Vaca, University of Seville, Spain
Ravi Jhawar, Università degli Studi di Milano, Italy
Dan Jiang, Philips Research Asia Shanghai, China
Georgios Kambourakis, University of the Aegean, Greece
Florian Kammueller, Middlesex University - London, UK
Sokratis K. Katsikas, University of Piraeus, Greece
Seah Boon Keong, MIMOS Berhad, Malaysia
Sylvia Kierkegaard, IAITL-International Association of IT Lawyers, Denmark
Marc-Olivier Killijian, LAAS-CNRS, France
Hyunsung Kim, Kyungil University, Korea
Ah-Lian Kor, Leeds Metropolitan University, UK
Evangelos Kranakis, Carleton University - Ottawa, Canada

Lam-for Kwok, City University of Hong Kong, Hong Kong
Jean-Francois Lalande, ENSI de Bourges, France
Gyungho Lee, Korea University, South Korea
Clement Leung, Hong Kong Baptist University, Kowloon, Hong Kong
Diego Liberati, Italian National Research Council, Italy
Giovanni Livraga, Università degli Studi di Milano, Italy
Gui Lu Long, Tsinghua University, China
Jia-Ning Luo, Ming Chuan University, Taiwan
Thomas Margoni, University of Western Ontario, Canada
Rivalino Matias Jr ., Federal University of Uberlandia, Brazil
Manuel Mazzara, UNU-IIST, Macau / Newcastle University, UK
Carla Merkle Westphall, Federal University of Santa Catarina (UFSC), Brazil
Ajaz H. Mir, National Institute of Technology, Srinagar, India
Jose Manuel Moya, Technical University of Madrid, Spain
Leonardo Mostarda, Middlesex University, UK
Jogesh K. Muppala, The Hong Kong University of Science and Technology, Hong Kong
Syed Naqvi, CETIC (Centre d'Excellence en Technologies de l'Information et de la Communication),Belgium
Sarmistha Neogy, Jadavpur University, India
Mats Neovius, Åbo Akademi University, Finland
Jason R.C. Nurse, University of Oxford, UK
Peter Parycek, Donau-Universität Krems, Austria
Konstantinos Patsakis, Rovira i Virgili University, Spain
João Paulo Barraca, University of Aveiro, Portugal
Sergio Pozo Hidalgo, University of Seville, Spain
Vladimir Privman, Clarkson University, USA
Yong Man Ro, KAIST (Korea advanced Institute of Science and Technology), Korea
Rodrigo Roman Castro, Institute for Infocomm Research (Member of A*STAR), Singapore
Heiko Roßnagel, Fraunhofer Institute for Industrial Engineering IAO, Germany
Claus-Peter Rückemann, Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster / North-German Supercomputing Alliance, Germany
Antonio Ruiz Martinez, University of Murcia, Spain
Paul Sant, University of Bedfordshire, UK
Reijo Savola, VTT Technical Research Centre of Finland, Finland
Peter Schartner, University of Klagenfurt, Austria
Alireza Shameli Sendi, Ecole Polytechnique de Montreal, Canada
Dimitrios Serpanos, Univ. of Patras and ISI/RC ATHENA, Greece
Pedro Sousa, University of Minho, Portugal
George Spanoudakis, City University London, UK
Vladimir Stantchev, Institute of Information Systems, SRH University Berlin, Germany
Lars Strand, Nofas, Norway
Young-Joo Suh, Pohang University of Science and Technology (POSTECH), Korea

Jani Suomalainen, VTT Technical Research Centre of Finland, Finland
Enrico Thomae, Ruhr-University Bochum, Germany
Tony Thomas, Indian Institute of Information Technology and Management - Kerala, India
Panagiotis Trimintzios, ENISA, EU
Peter Tröger, Hasso Plattner Institute, University of Potsdam, Germany
Simon Tsang, Applied Communication Sciences, USA
Marco Vallini, Politecnico di Torino, Italy
Bruno Vavala, Carnegie Mellon University, USA
Mthulisi Velempini, North-West University, South Africa
Miroslav Velev, Aries Design Automation, USA
Salvador E. Venegas-Andraca, Tecnológico de Monterrey / Texia, SA de CV, Mexico
Szu-Chi Wang, National Cheng Kung University, Tainan City, Taiwan R.O.C.
Piyi Yang, University of Shanghai for Science and Technology, P. R. China
Rong Yang, Western Kentucky University , USA
Hee Yong Youn, Sungkyunkwan University, Korea
Bruno Bogaz Zarpelao, State University of Londrina (UEL), Brazil
Wenbing Zhao, Cleveland State University, USA

International Journal on Advances in Security
Volume 8, Numbers 1 & 2, 2015

CONTENTS

Gunter Saake, Otto-von-Guericke University Magdeburg, Germany
Claus Vielhauer, Brandenburg University of Applied Sciences, Germany

Improvement of User Profiling, Call Destination Profiling and Behavior Pattern Recognition Approaches for Telephony Toll Fraud Detection

Anton Wiens, Sandra Kübler, Torsten Wiens, and Michael Massoth
Department of Computer Science
Hochschule Darmstadt – University of Applied Science
Darmstadt, Germany
e-mail: {anton.wiens | sandra.kuebler | torsten.wiens | michael.massoth}@h-da.de

***Abstract* — Phone fraud attacks cause a massive loss in the telecommunication sector every year. The internet raises new potentials for these attacks. Attackers can use the internet to get illegal access to telecommunication devices such as Voice over Internet Protocol (VoIP) phones and use them for fraudulent calls to expensive destinations in overseas countries. The generated financial damage affects the attacked customers and also the telecommunications companies that provide the telecommunication services. Especially, attacks on small and medium-sized enterprises can threaten their existence. This demands protection for the customers and companies by a fraud detection system with intelligent detection techniques for detection and prevention of these fraud cases. In this work, we present two statistical online user profiling approaches, as well as a call destination profiling approach and a behavior pattern recognition approach for toll fraud detection. All four approaches show good and promising results. Especially, the results of the call destination profiling, for detection of distributed attacks on a single destination, and the behavior pattern recognition, for detection of change in a user's behavior patterns, are promising for future work.**

Keywords—Fraud detection; telephony; CDR; behavior profiling; statistical.

I. INTRODUCTION

This paper is an extended version of [1]. The previous work got improved and combined with other works, which were derived from previous work [2] [3] of the authors for better results. We enrich this paper with new information about the previous work, subsequent improvements and derivations of it. Therefore, an initial user profiling approach, a new user profiling approach, a call destination profiling approach and a behavior pattern recognition approach are presented that try to solve the problems described in the following paragraphs.

Today's voice communication by Voice over IP (VoIP) mostly uses the internet for data transport. There are the drawbacks that the internet can basically be accessed by anyone, and that it links anyone to anyone. For example, it is possible for third parties with criminal intent to access private branch exchange (PBX) systems connected to the internet.

Fraudsters may have multiple options to abuse these systems. Systems that are insufficiently secured may be tapped. Access data that has been saved in these systems could be used to abuse, compromise or even gain full access to the whole PBX. If the PBX system has been taken over, a fraudster will be able to conduct telephone calls to premium rate service numbers or comparable call destinations, generating profit. The resulting cost, on the other hand, will often be charged to the victim or its telecommunication service provider, because of a general rule in telecommunication service providing called "Calling Party Pays".

The Communications Fraud Control Association (CFCA) reports losses of about 46 billion USD caused by telecommunication fraud in 2013, an increase by 15% compared to 2011 [4]. Not only financial damage is a problem caused by fraud attacks. Small providers may also suffer from reputation losses, causing customers to change the provider because of decreased trust and fear of repeated fraud attempts in the future.

To detect and counter these attacks, respectively fraud attempts, fraud detection systems are used. Often, these systems apply methods based on the generation of statistical profiles for each user. User profiles are generated that describe their behavior. These profiles will then be used as input for machine learning techniques, allowing for the detection of fraud [5] [6] [7] [8].

The German company "Deutsche Telekom" reported a huge success in the prevention of fraud cases with potential damages of about 200 million Euro, using an automated fraud detection system [9]. The research project "Trusted Telephony" at the University of Applied Sciences Darmstadt, from which the work at hand originates, pursues the goal to increase security in VoIP telephony, cooperating with a German telecom service provider. A key objective of the project is the development of a fraud detection system.

Recently, fraud cases were caused by security exploits in FRITZ!Box hardware (from the company AVM GmbH), which is often used in Germany [10] [11]. A FRITZ!Box is an integrated, multifunctional routing device, offering internet connectivity, VoIP capabilities and other services in local area networks. Such a unit is very popular in Germany. Because of the large amount of units in use, there is an increased risk in case of security vulnerabilities, especially for private users.

On the other hand, an exploitation of the recently disclosed security vulnerability of such a unit is only one possibility to start such attacks. The security vulnerability has been patched by the manufacturer in the meantime, but in the future, comparable vulnerabilities in similar hardware could turn up.

Therefore, it is important to be able to detect these cases and devise measures to counter them.

Figure 1. Depiction of the generation of call detail records and a detection system using them.

A. Call detail records

The data being analyzed in this work comprises fraud attacks that have been enabled by the occurred and already patched security vulnerability of the FRITZ!Box units [10].

A CDR is a text file containing all parameters of single telephone calls. Each CDR is written by a primary VoIP routing system called TELES.iSWITCH [12] as calls are set up (see Figure 1). CDRs contain information on caller, callee, call duration, starting time, as well as technical network parameters.

B. Structure of the paper

The structure of this paper is as follows: The related work is presented in Section II, followed by an explanation of behavior profiling in Section III, as it is important for the described approaches. Section IV gives an in-depth analysis of attacks on FRITZ!Box units. The following sections except future work and conclusion provide a description of various approaches, each followed by an experimental setup including results and a short conclusion. Section V is about the previous basic user profiling approach, which is being extended by the work at hand. In Section VI, a new basic behavior profiling concept is presented. In order to adapt to the FRITZ!Box incident and to detect these fraud cases, another approach by the authors called destination profiling is described in Section VII. The concept of communication behavior patterns, which also deals with the FRITZ!Box incident, is outlined in Section VIII. An overall conclusion is presented in Section IX, followed by future work in Section X.

II. Related work

This paper is an extended version of [1] and also includes intermediate works [2] [3] that improved the quality of presented techniques greatly.

The first intermediate work presented a behavior profiling different from user profiling to cope with distributed single target toll fraud attacks. The second intermediate work introduces behavior pattern recognition to detect changes in patterns of the users. Patterns can be defined and extended dynamically.

All of the previous and intermediate work is improved in this paper.

In [1], a method for toll fraud detection using statistical user profiling has been described, which can especially be applied when no significant amount of training data is available. Additionally, the method can be run in a mostly autonomous way, requiring only a minimum amount of external administration. The method applies two user profiles, one for a past period of time and one for a present period of time, each containing statistical features. The profiles are used to identify suspicious deviations of the user's behavior, by which toll fraud attempts are detected. In this work, the attacks on FRITZ!Box hardware and the possibility to detect these using the presented method had already been mentioned.

In the work at hand, the method from the preliminary work is adapted more closely to this attack pattern. The new method again uses two profiles of statistical features for each user, but differing in contents and their actual use for the detection of attacks.

Furthermore, other related work also describes different methods of user profiling for the detection and prevention of toll fraud in VoIP telecommunication [5] [6] [8] [13] [14] [15]. In contrast to this work, the previous work [2] does not apply simple user profiles, but a new kind of profile specified as Call Destination Profile. These profiles are used to characterize the behavior of a destination telephone number instead of a user's behavior. It is intended to detect special kinds of attacks this way.

These attacks cannot be detected with user profiling techniques alone and hence would go undetected if the method from [1] was applied.

As a means to visualize user accounts, self-organizing maps (SOM) are used in [16]. This visualization is used to differentiate between normal and fraudulent ones. The features *call destination*, *call start time* and *call duration* are extracted from the CDR data and used for analysis. According to the authors, the method has a true positive rate (TPR) of 90% and a false positive rate (FPR) of 10%.

A framework for self-organizing maps has been developed by Hollmén, Tresp and Simula [17] to cluster probabilistic models. User profiles using data of mobile communication networks have been used for test runs of the system. The output is presented visually, so that the fraudulent calls can be distinguished from normal ones.

The authors of [18] focus on the detection of superimposed fraud using two signature methods, each summarizing a user's behavior. The first presented approach is based on a deviation of the user's current behavior and his signature, while the second is based on a dynamic clustering analysis. In the second approach, a sudden change or "shift" of a user's signature from one cluster to another is the criterion for a classification as fraud. The similarity between a signature and a cluster centroid, which in itself is defined as a signature, is crucial for such a shift. The detection rates of both methods have been estimated: The first one promises a TPR of 75% and the second one a TPR of 91%. Also, a combination of both approaches is examined.

The framework *SUNsHINE*, which is able to detect and prevent VoIP fraud by combining real-time capable components with an offline statistical analysis, is presented in [19]. Multiple data sources,

network traffic data and CDRs, can be used. Different algorithms and techniques are used, e.g., rule sets, profiling, neural networks and clustering. No estimations concerning the detection rate are given.

The intermediate work [3] has been inspired by the concept of clustering algorithms, as the aspect of finding similarities has been adopted, leading to a different point of view in contrast to [2].

III. Behavior Profiling

The term "behavior profiling" describes a technique for differential analysis where the behavior of a given object is represented by a statistical profile. In literature, a distinction is made between absolute and differential analysis.

An absolute analysis examines a whole set of data, trying to identify fraud cases, but does not consider different types of user behavior. A call that may be treated as a fraud case for one user could be no fraud case for another user. For example, one user only makes long calls to his family at weekends and the other user only makes long calls to his family at workdays. If an absolute analysis considers long calls at workdays as fraud cases, the latter user will be considered as fraudulent, just because his normal behavior does not comply with the definition of normal behavior given by the other user. This problem can be avoided by looking at each user and his behavior differently, thus called differential analysis.

Differential analysis is preferred to absolute analysis in most of the related work, e.g., [20] [7] [21] [22]. The main argument is the ability of differential analysis to include the absolute analysis. In other words, a fraud case detected by an absolute analysis can also be found by a differential analysis, but a fraud case detected by a differential analysis cannot always be found by an absolute analysis [20].

In the profile, data from the object is accumulated, which is then used to generate statistics that describe the object's behavior, which are called features. Often, behavior profiles are applied in the form of user profiles [5] [6] [8] [13] [14] [15]. In most cases, a differential analysis is preferred over an absolute analysis. This is because the absolute analysis is a subset of the differential analysis [9].

For example, three variants of user profiling methods are presented in [7]. In this work, the parameters *duration per call*, *number of calls per customer* and *costs per call* are arranged in different ways into the groups *national calls*, *international calls* and *mobile calls.* These are used to generate statistics for the profiles.

User profiles are utilized to describe the behavior of users in the present and in the past, enabling a comparison of behavioral patterns. By this comparison, it is possible to detect suspicious fluctuations. These are analyzed in the next step in order to generate a decision on fraudulent or non-fraudulent behavior.

IV. Analysis of attacks on FRITZ!Boxes

The recent attacks at (and by) FRITZ!Boxes can be divided in two categories. The first category comprises the hostile take-over of a FRITZ!Box by exploiting a security vulnerability in its firmware. The second category comprises possible results of such a take-over, especially secondary attacks that are enabled by then remotely controllable units. Both categories are described in more detail in the following subsections. It is important to note that the initially possible attacks on these units cannot be conducted anymore, since the firmware has been updated by the manufacturer in the meantime [11]. The focus of the work at hand is at the possibility of fraud attacks on telecommunication systems by utilizing taken over secondary hardware, which is not unlikely to happen again in the future, and detecting it.

A. Primary hostile take-over of a FRITZ!Box

The basic idea to perform a hostile take-over of a FRITZ!Box was as follows: An attacker would set up a web site, which is to be visited by potential victims. The attacker would then be able to exploit the known security vulnerability of the FRITZ!Box in order to extract the master password. Using this password, the attacker would be able to access the command shell. Once this is done, the attacker could then deploy system commands, e.g., to make the unit call premium rate service numbers at the cost of the unit's owner [11].

B. Secondary attacks after the take-over

Attack attempts on other systems that had been conducted using taken over FRITZ!Boxes seem to be very similar in their basic approach. For an in-depth analysis, anonymized data on such attack attempts has been provided by a telecom company. The data being used is in accordance to the Federal German Data Protection Act (Bundesdatenschutzgesetz) [23]. All results from this analysis are based on this data and may not represent attack patterns that appeared at other telecommunication providers.

From a single user's view

From the perspective of a single user, an attack attempt may look as follows: An attacker tries to set up a call to a premium rate service number or a comparably expensive call destination, possibly also in another country. This is done multiple times during a short time span. As soon as the attacker has successfully set up a call to a given number, he will try to call this number again, as often as possible, and also in a short period of time. If the call attempts fail (e.g., because the number is not available), the attacker will try another number.

The difficulty to detect such attack attempts lies in the low frequency and the low duration of these calls seen from a single user's point of view. Attackers will avoid a detection using these two parameters by applying an approach described in the next section.

Exploiting multiple users

By exploiting the security vulnerability at multiple victims' FRITZ!Box units, attackers are able to hide their attack attempts neatly. The attack attempts are distributed across multiple taken over units. So, it becomes possible to mask obvious evidence of attack attempts, such as frequency and duration of calls. This will be illustrated by the following examples:

1. Attacker A conducts a hostile take-over of victim C and causes C's unit to start 30 calls to destination number B. The duration of each call is 20 seconds.
2. Attacker A conducts a hostile take-over of victim C and causes C's unit to start 5 calls to destination number B. The duration of each call is 5 minutes.
3. Attacker A conducts hostile take-overs of 30 victims and causes each victim's unit to conduct one call to destination number B. The duration of each call is 20 seconds.

In the first example, the attack at victim C can be detected by the frequency of the calls. In the second example, the attack can be detected by the extraordinarily long duration of the calls. In the third example, the features used before cannot be used again. Figure 2 shows a depiction of example three with just two victims of an attacker calling a premium service number.

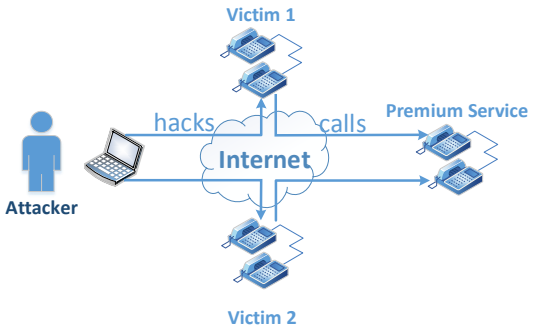


Figure 2. Depiction of example three with two victims of an attacker calling a premium service.

Existing methods often apply user profiling to detect suspicious behavior and potential attack or fraud cases. This way, distributed attacks, as described in the third example, cannot be detected. Therefore, it is necessary to apply a different method for detection.

C. Characteristic traits for detection

From the results of the preceding section, the following characteristic traits for detection can be deduced:

- **Duration of call for a certain user**: The call duration is significantly higher in comparison with the known behavior of that user.
- **Number of calls for a certain user**: The number of calls in a given time span is significantly higher in comparison with the known behavior of that user.
- **Number of calls for a certain destination number:** The number of calls that have been conducted to a given (premium rate service) destination number in a given time span is suspicious.

The first two of these characteristic traits can be detected by applying user profiling if the perspective of a single user is applied. To be able to detect attack attempts using the number of calls, a new method has to be devised. This will be described in Section VI.

V. PREVIOUS BASIC USER PROFILING APPROACH

The description and improved results of our previous user profiling approach will be provided in this section for completeness. The improvements to this concept are described in the sections of the new concepts following this section.

A. Constructing user profiles

For each user, two user profiles exist that represent the present and past behavior in specified time spans. The profile describing the past is called Past Behavior Profile (PBP), and the one describing the present is called Current Behavior Profile (CBP). Each profile uses features, calculated from CDR data, to describe the user behavior in its time span.

Features

Features describe different aspects of a user's behavior. In the profiles, the feature vector shown in Table I was used:

TABLE I. FEATURE VECTOR USED FOR USER PROFILES [7]

Max Calls	Max Duration	Max Costs	Mean Calls	Mean Duration	Std Calls	Std Duration

These are the maximum values (Max) for calls per hour (Calls), the duration of a call and the cost of a call, the mean value (Mean) and standard deviation (Std) for the same CDR information, except the cost.

For those features, the start time, duration and cost information of a CDR are needed. The cost of a call is depending on the user agreement and is not given in a CDR. Therefore, an approximation of costs for a CDR was made, based on country code, number type (mobile or fixed-line) and duration.

These features were used because they delivered the best results in [7]. Many works use standard deviation and mean values of the number of calls and the duration of a call to describe the user's behavior. Some works also differentiate them into national, international or mobile [21] [7] [22].

Profile time span

Each profile P has a length l_P. The PBP additionally has an offset $d \neq 0$, describing the difference in time between the present and the PBP time span (see Figure 3). For a CDR to be included in a profile, it needs to meet the following rules (1) and (2) for the corresponding profile:

$$T_{cdr} < T_n - d \quad (1)$$

$$T_{cdr} \geq T_n - (l_P + d) \quad (2)$$

T_n is the present (n) time, and T_{cdr} is the time of the CDR. If a CDR meets these two rules, it is included in the features of the corresponding profile.

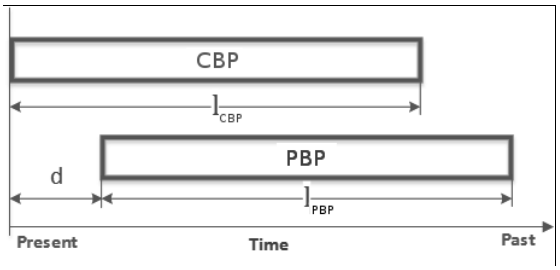


Figure 3. Profile time spans and offset (CBP = Current Behavior Profile; PBP = Past Behavior Profile).

The length (time span) of the profiles and the offset are very important parameters for the detection. The longer a profile is, the more CDRs are represented inside a profile and the statistics have more accuracy and less fluctuations. At the same time, the effects of single fraudulent CDRs become statistically more irrelevant and thus harder to detect. The offset is important for finding fraudulent CDRs that can only be found in groups. It decides how long it takes for a yet undetected fraudulent CDR to be included in the PBP and therefore makes it more unlikely to be found. The length of the offset also affects fluctuations when comparing both profiles. A higher offset causes higher fluctuations, a lower offset causes lower fluctuations likewise.

An optimal tradeoff between the length of the profiles and the offset between profiles needs to be found for best results.

Filling profiles

At first, the profiles need to get filled up for the method to be able to calculate meaningful features. Once the profile contains CDRs for its entire time span, the features can be calculated and used for further analysis. This means that the method has a determined training time for accumulating CDRs that is autonomously done without administration by personnel. In the following, a profile that has been filled up once is called *ready*.

B. Measuring change in user behavior

Once the profiles of a user are *ready*, the change of behavior measured by the profiles can be calculated. This is done by calculating the relative ratio R_F between each feature F of both profiles (PBP and CBP) by (3):

$$\forall F : R_F = \begin{cases} \left(1 - \left(\frac{F_{PBP}}{F_{CBP}}\right)\right), & F_{PBP} \leq F_{CBP} \\ \left(1 - \left(\frac{F_{CBP}}{F_{PBP}}\right)\right), & else \end{cases} \quad (3)$$

This results in a ratio R_F for each feature F, describing the change in behavior for that feature. Each R_F has a range of -1 to 1, with -1 as a maximum decrease and 1 as a maximum increase in behavior measured by that feature.

A ratio R_F for a feature F gives a relative value to the past behavior. It is relative because the severity of a change in user behavior is always relative to the past behavior of the user.

Empty profile

In case of a user not having made calls for a time span greater than the span of all user-specific profiles, one of the profiles of a user can run empty. Once a profile is empty, the calculation of the features is not possible, because they attain a value of zero. Comparing a non-empty profile with an empty profile will result in infinite ratios for the features, allowing for detection of fraud where there is none (e.g., when the PBP is empty and the CBP is not empty). Instead of letting the profile run empty, the last CDR in a profile that is about to become empty is not removed. This prevents the features from getting zero values and keeps user-specific information for fraud detection. Setting the features to a standard value would disregard user-specific behavior and is therefore not done.

Features accepting zero

Features like standard deviation can attain a value of zero, even if the profile is not empty. For example, the standard deviation of the duration attains a value of zero, if all calls in the profile have the same duration. Like in an empty profile, zero values are a problem for calculating the ratios. Therefore, a value ε (depending on the range of the specific feature) is added to the affected feature in both profiles.

C. Detecting fraud

For this approach, fraud cases are to be distinguished by extreme changes in user behavior described by each feature. Thus, for each ratio R_F of a feature F, a limit L_F is introduced. Each ratio R_F is checked if its limit L_F is exceeded, and the number (n) of exceeded limits is checked against an additional limit L_E (E for exceedings). If the limit L_E is exceeded, the CDR is labeled as fraudulent and as non-fraudulent otherwise. The procedure can be described as follows:

1. Set $n := 0$
2. $\forall R_F \in R: (R_F > L_F) \rightarrow (n = n + 1)$
3. $result = \begin{cases} fraud, & n > \mathrm{L_E} \\ normal, & else \end{cases}$

Once a CDR in the CBP is labeled as fraudulent, it is to be excluded from inclusion into the PBP. This prevents the PBP from including fraud cases and obscures potential follow-ups of fraudulent CDRs. This is the first approach chosen for a first experiment. Other approaches for detection using the ratios are discussed in future work.

D. Unexpected fluctuations

Many fluctuations in data and ratios, like weekends and holidays, can be predicted and adjusted for. But there are also fluctuations caused by random events inside the telecom service provider's network, e.g., network, hardware or other failures.

Those fluctuations are hard to predict using user profiles. The idea is to use the relation between absolute and differential analysis. If it is a fluctuation caused by the specific user, the fluctuation is not seen in an absolute analysis. If the fluctuation is global, it will affect all users and will be seen for specific users, too. Therefore, the accumulated behavior of all users has to be measured to detect this kind of fluctuation.

Because the functionality to measure user behavior has already been defined, it can be reused to measure the accumulated user behavior. A global version of a CBP and a PBP is needed for all users. Ratios are calculated the same way as in user profiles. In this case, the ratios are not used for fraud detection, because the source of the fraud cannot be detected by creating profiles for all users. The ratios are used to be included in the user-specific ratios for finding the global fluctuations and removing them from user fluctuations.

The inverse ratios of the global profiles are taken to the power of g and are multiplied with the corresponding ratio of a specific user profile as in (4):

$$newratio = (\mathbf{1} - globalratio)^{g} \cdot userratio \qquad (4)$$

An appropriate value for g is determined in Section V.F. Both ratios have the same scaling and global ratio that describes the change for the user ratio that is still normal. Therefore, the inverse is multiplied by the user ratio. Because the global ratio is much more stable with more samples, it is taken to the power of g. g is dependent on the scaling of $globalratio$ and not on $userratio$.

E. Low usage users

An analysis of the data revealed that on average, each user only makes 6-7 outgoing calls per day. About 47% of the users only conduct 2 calls per day on average. That means a lot of users — and therefore user profiles — include low amounts of calls. Hence, only few samples are available for calculating the statistics, making the statistics inaccurate. A way to handle those fluctuations is to scale the calculated ratios for the user by the number of samples inside the profiles. For the creation of a scaling function $S(x)$, the dependencies of the number of calls in the profiles and the ratios needed to be analyzed.

Before and after scaling a ratio, it needs to be converted to linear space with (5).

$$S(x, y) = 1 - \frac{1}{\left(\left(\frac{1}{1-y} - 1\right) * S(x)\right) + 1} \qquad (5)$$

x is the number of calls in the PBP, and y is the ratio to be scaled. The part $\left(\frac{1}{1-y} - 1\right)$ scales the ratio into linear space, and $1 - \frac{1}{(\dots) + 1}$ reverts it back to the previous space. A full overview of all components and their relationships is shown in Figure 4.

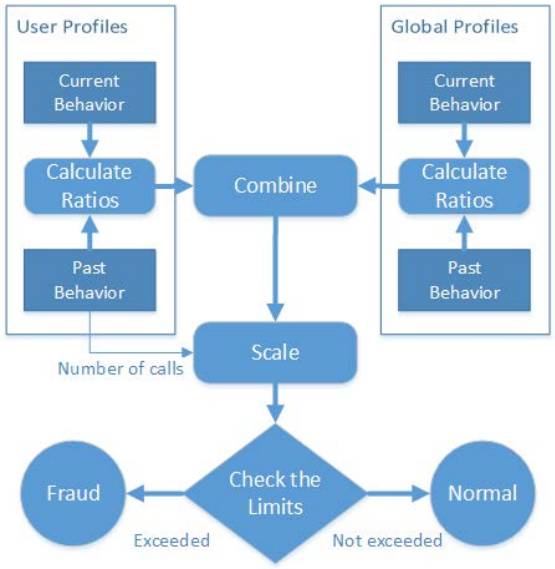


Figure 4. Overview of the components and their relationships.

F. Results of the previous approach

This section describes the test of a prototype implementation of our previous approach in an experiment. The implementation has been done in Java for an existing fraud detection framework of the research project. The data used for the experiment has been generated by a live environment, recorded by a VoIP switching device. The data consists of 76,326 cost impending calls and spans over a time of one month. It has been anonymized in accordance to the German Federal Law on Data Protection.

For the experiment, the whole data set was used, as the system trains on live data with the assumption that fraud cases are rare enough so that the profiles can initially be trained by themselves without greater risks of being manipulated by fraud cases. Assuming the contrary is true and the first data set is containing fraudulent CDRs, the impact would only be that no fraud cases are detected until the fraudulent CDRs are no longer used for the PBP.

For the experiment, profiles of a week's length and with an offset (d) of one day for the PBP are used. In a first run, all occurring ratios are recorded to calculate limits for the ratios, to analyze the parameters for the scaling function and to integrate the global ratios into user profiles. In a second run, the limits were applied and the fraud detection component was enabled.

First results

For the first results, without incorporating the global profiles and the scaling function, the false positive rates (FPR) for different limits were measured. The false positive rate is a very important measure that indirectly determines the expenses due to inefficiency, because administrators need to look at false positives.

Table II shows empirically tested limits for ratios and the number of exceedings. The FPR has been measured from 50,893 samples, where the profiles were *ready*. The limits and the resulting FPRs will be used for comparison with results of the

TABLE II. FIRST RESULTS OF FPR WITHOUT GLOBAL PROFILES AND SCALING FOR DIFFERENT LIMITS WITH PROFILE LENGTH OF ONE DAY AND AN OFFSET OF ONE DAY

Limit for all ratios	Limit for exceedings	FPR
0.25	>0	0.2142
0.25	>1	0.1274
0.5	>0	0.0685
0.5	>1	0.0444
0.75	>0	0.0211
0.75	>1	0.0145

incorporations of global profiles and the scaling function for low usage. These results are for CBPs length of a day and the PBP length of a week.

The following tables show additional information about other time spans for the profiles in comparison to already shown results.

As seen in Table III and in Table IV, the results show that a profile's length of one week with an offset of one day is more promising with much lower FPR than the other profile variants.

TABLE III. RESULTS OF FPR WITHOUT GLOBAL PROFILES AND SCALING FOR DIFFERENT LIMITS WITH PROFILE LENGTH OF ONE DAY AND AN OFFSET OF ONE DAY

Limit for all ratios	Limit for exceedings	FPR
0.25	>0	0.7274
0.25	>1	0.6355
0.5	>0	0.5066
0.5	>1	0.4283
0.75	>0	0.3024
0.75	>1	0.2371

TABLE IV. RESULTS OF FPR WITHOUT GLOBAL PROFILES AND SCALING FOR DIFFERENT LIMITS WITH PROFILE LENGTH OF ONE WEEK AND AN OFFSET OF ONE WEEK

Limit for all ratios	Limit for exceedings	FPR
0.25	>0	0.5964
0.25	>1	0.4467
0.5	>0	0.3096
0.5	>1	0.2239
0.75	>0	0.1309
0.75	>1	0.0386

Global profiles

For the global profiles, the same length and offset was used, because the ratios can be compared better if the parameters are similar. The number of calls was used as the only feature for the global profiles. For the parameter g for scaling the global ratio, see (4), a test value of 1 was used.

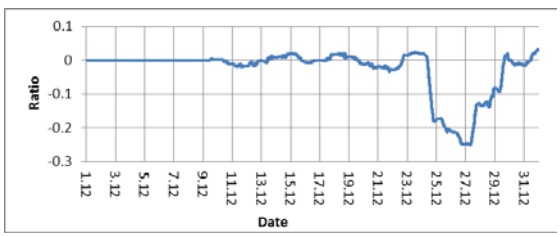


Figure 5. Ratios for number of calls for the whole data in global profiles.

Figure 5 shows the ratios measured for the given data, chronologically sorted. It shows negative ratios during the Christmas holidays in Germany, successfully measuring its effects on the ratios and it can be used to remove those effects from single user behavior. Also, this figure shows when the profiles became *ready*.

The incorporation into profiles of a week's length showed no significant improvements in the FPRs.

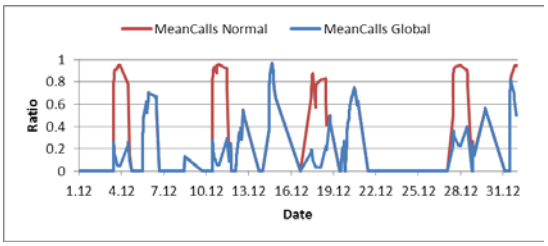


Figure 6. Example incorporation of global ratio into a day-length user profile for feature MeanCalls.

On the other hand, a small scale test of profiles with a day's length showed very good results in removing weekend fluctuations from the profiles. Figure 6 depicts an example for day-length profiles.

The figure shows two curves, MeanCalls Normal showing the ratios of the feature MeanCalls without correction by global profiles and MeanCalls Global with correction by global profiles.

Scaling for low usage

To find an appropriate scaling function, the dependency of the number of calls to the maximum occurring ratios was analyzed. Figure 7 shows an example for four features. It depicts how a low number of samples/calls in a profile can affect the ratios. Therefore, a scaling function was created that scaled the ratios from 0 to 70 calls.

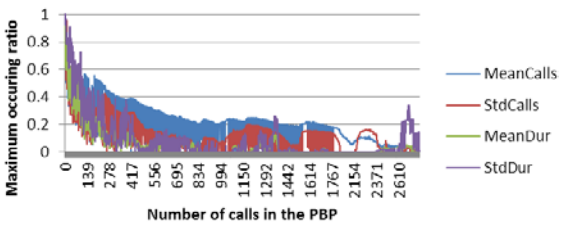


Figure 7. Example for the dependency of max values of the features MeanCalls, StdCalls, MeanDur and StdDur to the number of calls.

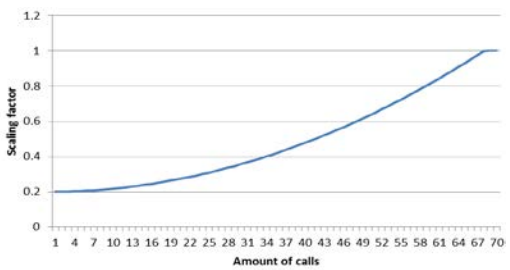


Figure 8. Depiction of the scaling function.

For the scaling function, a simple parable of the form $y = (ax)^2 + b$ was chosen after testing different curves, because it corresponds well to the curve in Figure 7. Using the coefficients $a = \frac{1}{67.1}$ and $b = 0.2$, the scaling begins at 0.2 with 0 calls

and ends at 1 with about 70 calls with a slight increase, as shown in Figure 8. Because about 47% of users only conduct about two calls per day, the scaling function greatly improved the FPRs, as shown in Table V.

TABLE V. CHANGES IN FPR WITH INCORPORATION OF THE SCALING FUNCTION

Limit for all ratios	Limit for exceedings	Old FPR	New FPR	Change in %
0.25	>0	0.2139	0.1684	-21,27%
0.25	>1	0.1272	0.0939	-26,17%
0.5	>0	0.0683	0.0491	-28,11%
0.5	>1	0.0443	0.0290	-34,53%
0.75	>0	0.0211	0.0136	-35,54%
0.75	>1	0.0145	0.0083	-42,75%

Determination of limits

The best way to determine the limits is to optimize the ratio of true positive rate to false positive rate. However, this requires labeled data to be possible. Because of the lack of labeled data, the limits were determined by measuring the 99.5% quantile of all occurring ratios for each feature. The ratios are presented in Table VI. Using these limits, the measured FPR is 1.87%.

TABLE VI. LIMITS FOR FEATURES (99.5% QUANTILE)

Feature	Limit
MaxCalls	0.8247
MaxDur	0.6692
MeanCalls	0.7512
StdCalls	0.8270
MeanDur	0.2985
StdDur	0.5400
MaxCost	0.7387
Mean	0.3835

Final detection rates

Out of the 50,893 analyzed cost impending calls, 1.87% are measured as false positives. Through empirical inspection of the false positives, two users were found with an exceptionally strange behavior pattern. The duration of calls and the number of calls per second was the same in about 200 calls, which is very suspicious. After consultation with the providing telecom company, those calls were considered fraud cases. This shows that the presented approach can detect false positives and reduce the FPR to 1.22%, but does not provide a true positive rate for a decent comparison with related work. Still, 90.23% of the fraudulent calls found in these two users were marked as fraud by the proposed approach. Compared to the approach proposed in [24], which also proposes a statistical, unsupervised method, the approach of this paper has a lower FPR (1.22% to 4.0%). Compared to other supervised techniques, like [13] (with 50% TPR and 0.3% FPR) or [22] (two approaches with 70% and 80% TPR and 0% FPR for both), the proposed approach has a good TPR and FPR and needs no effort for preparing supervised training data.

VI. NEW BASIC BEHAVIOR PROFILING CONCEPT

In this section, a basic concept for behavior profiling is described, consisting of profiles, calculation of features for two different contexts and detection of anomalies in the profiles. The idea for this concept was derived from the previous concept described in Section V. The need for a new concept arose from the problems with low activity users and fluctuations described for the previous approach and its complexity, but also from the potentials for using this approach in a different context. In the following, a description of the new basic behavior profiling concept is given.

A. Profiles

A profile is a collection of historic data in the context of an object. Examples for objects are users or destinations. The historic data, for this work, consists of call information. A profile P holds the information that occurred in a time span with the length P_L with an relative offset P_O to the present time p.

To determine if data with a given timestamp t is inside the time span of a given profile P, the following rules are applied:

$$inside = \begin{cases} true, if\ (p - P_O - P_L) < t < (p - P_O) \\ false, else \end{cases} \quad (6)$$

If data is inside a profile's time span, it is used to calculate features that describe the behavior of an object in that time span. Profiles that describe the behavior of the present will be called Current Behavior Profile (CBP) and profiles describing past behavior will be called Past Behavior Profile (PBP).

B. User profiling

As mentioned in the introduction, we introduce two contexts based on previous work. The first context, as it is common for the related work, is the user's context. The second context is the destinations context.

For the user context, the outgoing call behavior of the user is analyzed to find deviations from the normal or present user behavior to the past behavior. These deviations or anomalies are used to detect fraud.

The new approach has not as much problems to be considered as the old approach. The idea of global user profiling from the old approach for global fluctuations in user behavior, e.g., due to seasonal reasons, got adapted to the new approach and is described in the following subsections.

Features and profiles

The most used features for describing past user behavior used in related work are statistics over the amount of calls and the duration of each call a user makes, e.g., in [5] [7] [20]. Therefore, the standard deviation and the arithmetical average of the call amount and the duration of calls are used in this work.

In contrast to the previous approach, the maxima are not used in this concept, because an analysis

showed weak influence in detection by these features.

The statistics for the duration will be calculated on per call basis and the statistics for the call amount on a per hour basis.

Table VII shows the used feature vector for describing the past behavior. Mean stands for arithmetical average, Std for standard deviation, DpC for Duration per Call, and CpH for Calls per Hour.

TABLE VII. FEATURE VECTOR USED FOR DESCRIBING PAST USER BEHAVIOR

MeanCpH	StdCpH	MeanDpC	StdDpC

From the CDRs, both the timestamp of the call connect and the duration are needed for calculating the features.

For the present user behavior, only the call amount and the average duration for the latest hour are used as features, resulting in the following feature vector described in Table VIII.

TABLE VIII. FEATURE VECTOR USED FOR DESCRIBING PRESENT USER BEHAVIOR

MeanCpH	MeanDpC

The CBP has a length P_L of one hour and no offset. Therefore, MeanCpH stands for the number of calls in the profile. The PBP has an offset P_O of one hour, due to the length of the CBP, and will have a length of one week as discussed in Section V.A and shown with good results in Section V.F.

Detection

The detection of fraud is done by comparing the PBP with the CBP features. For this, we calculate a limit for the mean duration and number of calls of the CBP. The limit is calculated as described in (7):

$$Limit = Mean + Std * r + a \qquad (7)$$

a is an additional absolute part that removes the need to handle users with low amount of calls and empty profiles. It needs to be small enough for not affecting users with a high call count and still protect users with a low call count from unnecessary fraud alerts. The techniques for these cases described and used in the previous work are therefore no longer needed. The Mean and Std part scales with the amount of calls the user does and is therefore a scaling for users with a higher amount of calls. The additional scaling r is for adjustment of the relative part.

Adaption of global user profiling

In the previous approach, we used a global user profiling for analysis of fluctuations in the data like:

- Holidays
- Seasonal fluctuations
- Weekly fluctuations (weekends)
- Daily fluctuations (work/after work)
- Unexpected fluctuation (network problems)

We used global user profiling with the assumption that the whole user base as a single entity provides stable enough statistics that show the global fluctuations but not the single user's fluctuations. This also requires that a single user does not make more calls than all other users together.

In this concept, because of the profiles' length of one week, we only want to look at holidays, seasonal fluctuations and unexpected fluctuations, e.g., caused by network failures. The other fluctuations get evened out by the statistics over one week.

In contrast to the previous approach, we will also use the user profiling provided in this concept for global user profiling. This will make it easier for changes in the global user profiling to be factored in the individual user's profiling.

For the PBP and CBP, the same lengths and features are used as for user profiling.

We measure ratios R for each feature F between the CBPs and PBPs. The ratio is calculated by dividing a feature of the CBP with the respective sum of the mean and standard deviation feature.

The ratios must then be applied to the detection of a single user's profiling. The relative part of the limit formula (8) is adjusted by the respective ratio:

$$Limit = (Mean + Std * r) * R + a \qquad (8)$$

C. First Results

A first analysis in "normal" data showed for single users that the limit is appropriately high enough. This was enabled by the incorporation of the global ratio. Figure 9 shows the difference between normal ratio of a single user and his ratio adjusted with the global ratio for the MeanCpH limit. The figure shows the information of increased activity during working hours that is normally lost in the user's profile due to the smoothing of the statistics over one week. This confirms that we can successfully reinsert an approximation of this information via the global ratio.

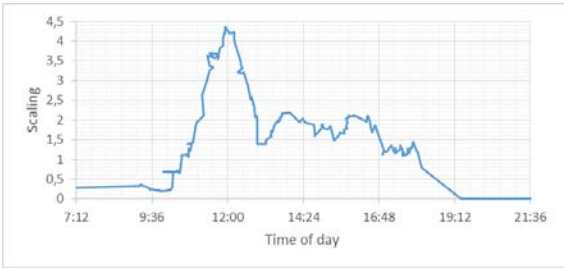


Figure 9. Depiction of the difference in limits with incorporation of global ratio for the MeanCpH limit.

This shows that the new approach is much simpler than the old one, but still handles the problems of the previous approach and promises good results for detection rates (false positive rate and true positive rate).

An extensive analysis over prepared and labeled data for acquiring detection rates of this new user profiling approach is still required.

VII. DESTINATION PROFILING

After the FRITZ!Box attacks happened, as described in Section IV, the previous approach was applied to a data set containing the attacks. The previous approach could not detect the FRITZ!Box attacks because the context of the profiling was that of a single user and not of a destination. Attacks on a single destination distributed over many users could therefore not be seen by user profiling, because of the relatively small effects on the single user's profile. This led to the idea of using a different context for profiling and detection of distributed attacks on single destinations, as we call it Destination Profiling.

In another previous work [2], Destination Profiling was then implemented and good results were generated for detecting this kind of attacks. The results are shown in Section VII.B after the description of the approach for detection is given in the following paragraphs.

Because of the good results in the previous work with profiling the number of distinct users calling a single destination in a defined time span [2], the approach is improved by adding distinct callers as a feature for the Destination Profiling. After the results of the previous work are shown, the new results with the improvement are shown and compared to the previous results.

A. Features, profiles and detection

For Destination Profiling, the same methods as for the User Profiling are not only reused, but adjusted. The duration of a call is not representative for a destination, because different callers have different behaviors. Only the number of calls, that is not per call basis, and the number of distinct callers can give important information about fraudulent usage of the destination.

Only the number of distinct callers can be used, because the number of non-distinct users will lead to the same result as with user profiling. If only one caller is doing a high amount of calls, this can also be detected with user profiling, as there is only one user context doing the calls.

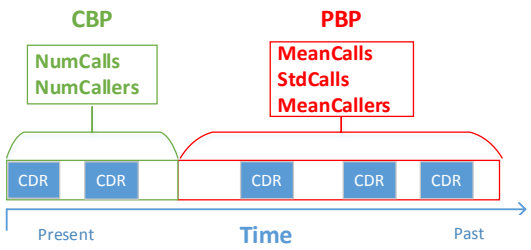


Figure 10. Destination Profiling: Depiction of the current behavior profile and the past behavior profile in relation to time.

For the PBP, the mean and std for calls per hour and callers per hour are used. For the CBP, the number of calls and the number of distinct callers are used. The profile parameters and the detection are the same as for User Profiling. Figure 10 depicts the profiles.

B. Previous prototype

In this section, results from a prototype implementation of the previous Destination Profiling approach are described and analyzed empirically.

Used Data Set

To evaluate the prototype implementation, real life traffic data (CDRs) provided by a local telecom company has been used. The data comprises calls from a time span of two weeks containing about 3.5 million calls. Only the portion of the data with outgoing calls was used, because incoming calls are not relevant to the analysis. The outgoing calls amount to about 470,000. Table IX shows the distribution of the calls for the regions national, mobile and international and are split into connected and unconnected calls.

TABLE IX. DESTINATION PROFILING: NUMBER OF CDRS FOR EACH REGION

REGION	AMOUNT
CONNECTED	325,947
NATIONAL	274,205
MOBILE	42,669
INTERNATIONAL	9,073
UNCONNECTED	153,330
NATIONAL	112,476
MOBILE	24,570
INTERNATIONAL	16,284
TOTAL	479,277

In the first week, no attack attempts (fraud) were contained. This part of the data was applied to initialize the behavior profiles, building the features. In the second week, normal call traffic is contained, as well as about 20,140 fraudulent calls following the typical FRITZ!Box attack pattern. The second week has been used to test the detection abilities.

Experimental Setup

First of all, the relevant thresholds had to be determined, because this is a necessity for high-quality detection results. To accomplish this, a single run of the method, without the fraud detection, is conducted with the first week of the data and every feature value at the time of each call is recorded. The thresholds are estimated by analyzing the resulting values of the CBP for fraud and non-fraud cases and for each region (national, mobile, international). The 99%-quantiles of the number of calls from the CBP, for connected and unconnected calls, as well as national, international and mobile calls each, have been recorded and used as the absolute threshold A_R for each region. The parameter G_R, representing the relative threshold, has been set to $G_R = 1$, for testing purposes.

Finally, a test run with the activated fraud detection and the previously measured thresholds is done and the detection quality is evaluated by

comparing the detected cases to the known cases of fraudulent behavior.

The approach can be described with the following steps:

1. The detection method is deactivated at first
2. The profiles are initialized using the data set of the first week
3. Thresholds are calculated from CBP values as described before
4. The detection method is now activated
5. The data set of the second week is now used as input
6. The results from the detection method are compared to the known cases of fraudulent behavior

Detection results

Thresholds have been determined for successfully connected, as well as unconnected call attempts, each for national, international and mobile calls. Also, the profile values have been calculated and recorded.

The arithmetic mean and the standard deviation both represent valid values to generate relative thresholds. An adjustment with the parameter G_R is only necessary in individual cases.

Under these testing conditions, the detection method achieved a false positive rate of 0.7% or 3,355 false positives (see Table X). Of the known attacks in the data, the detection method was able to identify all attacks, resulting in 100% detection rate or true positive rate. However, there is the possibility that not all attacks are detected because some may still be unknown to the provider of the data. An estimation of a true positive rate of about 95% would be more appropriate.

TABLE X. DESTINATION PROFILING: DETECTION RESULTS

	AMOUNT	RATE
FALSE POSITIVE	3,355	0.7%
TRUE POSITIVE	20,140	100%

Compared to the results achieved in comparable related work (see Table XI), which utilizes unsupervised user profiling, with a FPR of 4% and a TPR of 75% [6] and our previous user profiling approach with a FPR of 1.22% and a TPR of approximately 90% (see Section V), these measurements are as good or even better.

TABLE XI. DESTINATION PROFILING: COMPARISON OF FPR AND TPR

	TPR	FPR
THIS WORK	95%	0.7%
PREVIOUS APPROACH	90%	1.22%
RELATED WORK [6]	75%	4%

On the other hand, no direct comparison is possible, because the detection method itself is partially different, applying a modified approach of user profiling.

Improved prototype

The addition of the number of distinct users as a feature to the profiling allowed the reduction of the limit for number of calls while maintaining the true positive rate. This also affected the false positive rate and reduced it by 0.2%, resulting to a FPR of 0.5%.

VIII. CONCEPT OF COMMUNICATION BEHAVIOR PATTERNS

The concept of communication behavior patterns in one of the intermediate works [3] was developed based on the analysis of the FRITZ!Box incident and experiences with user profiling from the previous work. It differs in the *point of view* from the concept of Destination Profiling [2], utilizes pieces of information from user profiles and adapts the principle of clustering algorithms (unsupervised learning) [25], which is used to find similarities between objects.

Behavior patterns are created in order to reflect a behavior of a user in a specific context. To associate with a behavior pattern, each shall have its own criteria, where similar patterns possess similar criteria. In order to describe the behavior concerning a distinct aspect of a user or a group of users, the calls of a user profile are matched against predefined behavior patterns. A user is able to have matches to several behavior patterns.

To search for behavior patterns using user profiles as objects and to obtain an indication for thresholds, clustering algorithms implemented in WEKA [26] were used. Used algorithms were k-means, EM and an implementation of a SOM (self-organizing map) as a clustering algorithm.

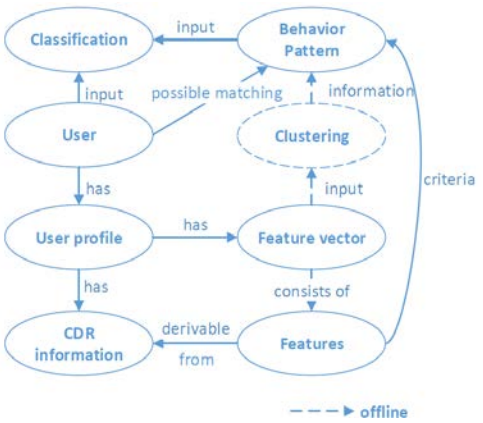


Figure 11. Overview of the relations between the components.

Figure 11 shows an overview of the relations between the components of the concept of behavior patterns. For every user, a user profile gets constructed. Each user profile consists of prepared data retrieved from CDRs, as well as a feature vector, which are defining criteria for behavior patterns. Clustering algorithms providing pieces of information for behavior patterns are used offline, as the gathered information has to be evaluated, as well as they potentially slow down the whole classification process. At last, using a user's behavior patterns and the belonging matching

information, a call can be classified as fraudulent or non-fraudulent.

A. Data preparation for user profiling

The concept of behavior patterns utilizes information retrieved from user profiles. The attributes A_1 to A_4 are extracted from a CDR for a user profile:

A_1 User ID
A_2 Timestamp of the call
A_3 Duration of the call
A_4 Destination number

The first attribute is used for a unique identification of a user. A_2 is used to obtain information whether the call occurred on a weekend and if the call has been made during work hours (7:00 am to 18:59 pm) or after hours (19:00 pm to 6:59 am) with a time span of 12 hours each.

A_3 is used to know whether the call was a call attempt or call connect. The information obtained from A_4 is further categorized into its call region *national*, *mobile* and *international*.

B. Behavior patterns

A behavior pattern reflects a behavior in a specific context of a user. An example is the behavior pattern "International Calls" to which a user gets assigned if he conducts calls to international destinations often. It is possible for a user to *match* one or more behavior patterns.

Features

Every behavior pattern has its own defining set of features *F* called *feature vector*, with comparable behavior patterns having similar defining features. These features are highly dependable on the criteria of a behavior pattern. Therefore, providing an overall definition for a feature vector is not possible. The features are derived from the data contained in a user profile and are grouped in two types, numeric and Boolean (true/false).

Examples for two behavior patterns, their criteria and therefore feature vectors are:

- "*International Calls After Hours"*: The criteria for this pattern are: The call has to be connected, the call region is *international* and the call is made *after hours.*
- "*Weekend Calls"*: The only criterion is for the calls to be made on a weekend.

The numerical value is the accumulation of the respective calls during a time span t_{BP} (in this case, one hour), which applies for both behavior patterns.

Criteria for a behavior pattern match

For a *match* to a behavior pattern, every feature of a feature vector, depending on its type, has to meet its criteria:

- Numeric: A statistical or numeric feature has to pass a *threshold*.
- Boolean: A Boolean feature has to have the value *true*.

For every defined behavior pattern, the criteria are tested. This way, it is possible for a CDR of a user to lead to a match to more than one behavior patterns.

Metric for a match

All calls matching a distinct behavior pattern are stored in respective lists. Over time, the length of such a list (*grade* of a match) can diminish or grow, being further denoted as a *growth* of a match to a behavior pattern.

The *growth G* of a match to a behavior pattern over a time span is measured as:

$$G = \frac{C_L}{\bar{x}\,(C_P)} \tag{9}$$

C_L denotes a list of all connected calls during the current (latest) hour and C_P a list of all connected calls in the past. For both C_L and C_P, calls from the list of matches are used. $\bar{x}$ denotes the arithmetic mean over the respective list.

Change of a match

The *growth G* of a match to a behavior pattern described above is further used as a criterion to mark a current call as *fraudulent*, as it is defined in the following case differentiation:

$$Fraud = \begin{cases} true, G > T_{BP} \\ false, otherwise \end{cases} \tag{10}$$

T_{BP} denotes a threshold for the growth of a match to a behavior pattern. If T_{BP} is passed, the current call, which had been causal for *passing* the threshold, is the first call to be considered fraudulent. All subsequent calls, which are still triggering *true*, are considered fraudulent as well. To regulate how much a growth of a match influences the assignment of a call as fraudulent, each behavior pattern has been given a *weight*, leading to an enhancement of the case differentiation shown in (10) to (11):

$$Fraud = \begin{cases} true, G \cdot w > T_{BP} \\ false, otherwise \end{cases} \tag{11}$$

C. Prototype

Used data

In case of the prototypical implementation of the concept of behavior patterns, real life traffic data over a time span of seven weeks provided by a local telecom company has been used. The first week has been used for the initialization phase, where user profiles, as well as behavior patterns of a user, are constructed, as this week did not contain known fraudulent activity. Out of the seven weeks, there is at least one week included with definite fraud attacks having the pattern described in Section IV. The rest of the data shows partial signs of the FRITZ!Box fraud attack pattern as well. The data set comprises 10,401,547 CDRs. As only outgoing calls, as well as successfully connected calls (call connects) are of importance, 2,749,860 CDRs were left.

Experimental setup

Two simple behavior patterns have been defined:

- *IntCallsPattern*: All connected calls having an international destination match the behavior pattern.
- *IntCallsAfterHoursPattern*: All connected calls having an international destination and having

been conducted in the after hours match this behavior pattern.

The thresholds for the statistical features for both behavior patterns, as well as indications about the thresholds concerning the change of a match to a behavior pattern have been derived using clustering algorithms from WEKA. The applied clustering algorithms were k-means, EM and an implementation of a SOM as a clustering algorithm. For the prototypical implementation, the behavior patterns possess parameters, which can be defined via a XML configuration. Table XII shows the definition of both behavior patterns, including the parameters. The parameter *type of pattern* defines whether the behavior pattern checks for the number of calls (*calls*) or for the sum of duration (*duration*). The possible values for each parameter after "type of pattern" are:

- call type (all / call attempts / call connects),
- destination (all/national/international/mobile),
- timeslot (all / workhour / after hour) and
- weekday (all / workday / weekend)

If no further distinction is made, a parameter gets initialized with *all*.

TABLE XII. OVERVIEW OF THE DEFINED BEHAVIOR PATTERNS AND THEIR PARAMETER VALUES

	BP_1	**BP_2**
name	IntCalls	IntCallsAfterHours
weight	0.9	0.7
threshold (matching)	25.2	8.4
threshold (growth)	0.5	0.4
type of pattern	calls	calls
call type	call connect	call connect
destination	international	international
timeslot	all	after hours
weekday	all	all

Results

An approximation concerning the TPR was possible due to the analysis performed on the data retrieved during the FRITZ!Box incident. It is highly possible that not all fraudulent data has been known during the evaluation of the prototype. The following steps have been applied on the data set:

1. Apply the thresholds and weight values retrieved from clustering algorithms and given from experience, respectively.
2. Run the prototype with the defined two behavior patterns.
3. Analyze the results utilizing the knowledge derived from the analysis of the data, as well as from the local telecom company.

In total, 17,110 fraud cases were reported and analyzed. During the analysis, one customer was noticeable in his behavior to conduct calls to foreign destinations very often, even not during the timeframe of the FRITZ!Box incident. Due to these findings, as well as other aspects found in our analysis, the aforementioned customer can be considered being a call center. As such customers are likely to be added to a whitelist, all calls belonging to such a customer can be ignored, which leads to a total of 13,503 reported fraud cases. A TPR of *98.4%* and a FPR of below 0.01 % have been measured. At this point, it has to be said that surely not all fraud instances of the FRITZ!Box incident could be found. This can be said even though not enough labeled data existed, as valuable time – and therefore, CDRs – passes in order for a user to match a behavior pattern and be associated with the described behavior. Additionally, a threshold concerning the growth of a match has to be passed, resulting in an equivalent to a "settling-in phase". Thus, it is possible that not all fraudulent instances were detected.

IX. CONCLUSION

This paper presents an already simple statistical way of detecting fraud in telephony by analyzing user behavior and finding anomalies. It has flaws concerning the complexity, because of problems with special cases of users and fluctuations. These problems and the need for a technique for detecting the FRITZ!Box attacks led to three new techniques.

With the new user profiling concept, the complexity of the previous approach is reduced and the problems of it are handled, by combining the global profiling with a different statistical approach. The effect of the global profiling is shown for a single user. Still, more features are needed to describe user behavior as it is shown in related work.

The FRITZ!Box attacks led to an adaption of the new user profiling approach to a new context for enabling the detection of these attacks, called Destination Profiling. Destination Profiling allows to detect attacks on a single destination from multiple sources. The developed approach shows promising detection rates and was improved with small changes in this paper.

The concept of behavior patterns utilizes the grouping aspect of clustering algorithms, leading to behavior pattern recognition using information retrieved from user profiles. Pieces of information from a user profile are matched against predefined behavior patterns, which depict the behavior of a user in a specific context. A match to a behavior pattern can grow and if a significant growth in a short time frame has been observed, a call is considered fraudulent.

Overall, the three new approaches that were derived from our previous work show promising results for a combined detection in an online analysis tool.

X. FUTURE WORK

The most important future work is an extensive analysis of all new approaches presented in this work. The analysis needs to be done on a large enough prepared and labeled data set. A very important task of this analysis is to find correlations between the approaches to create an optimized combined detection result.

In particular, the new user profiling approach needs more features for the description of user behavior. Features used in related work need to be further analyzed and adapted to this approach. The incorporation of the global profiling needs to be fine-tuned as well.

As there is little related work for destination profiling, a more detailed analysis of possible new features for the presented approach needs to be done.

The behavior pattern recognition only covers a little amount of possible user groups. The authors see a huge potential in finding new user groups and analyzing them for finding new approaches for fraud detection. Including information given by call attempts and call termination cause codes can further improve the detection result. They can provide insight whether a fraudulent attack is currently prepared or conducted. Additionally, “normal” behavior patterns - e.g., “National Calls” - have to be considered as well, as they can provide further indications on a sudden change in a user’s behavior.

ACKNOWLEDGMENT

We would like to thank the state of Hesse, Germany for supporting this work by providing the necessary funds by the development program LOEWE. Also we would like to thank a German telecom company that provided the necessary data, essential support and knowledge for research and development of practical fraud detection methods for this work.

REFERENCES

[1] A. Wiens, T. Wiens, and M. Massoth, "A new unsupervised user profiling approach for detecting toll fraud in VoIP networks," in The Tenth Advanced International Conference on Telecommunications (AICT 2014) IARIA, 2014, pp. 63-69.

[2] A. Wiens, T. Wiens, and M. Massoth, "Approach on fraud detection in Voice over IP networks using call destination profiling based on an analysis of recent attacks on FRITZ!Box units," in The Sixth International Conference on Emerging Network Intelligence (EMERGING 2014) IARIA, 2014, pp. 29-34.

[3] S. Kübler, M. Massoth, A. Wiens, and T. Wiens, "Toll fraud detection in Voice over IP networks using communication behavior patterns on unlabeled data," in The Fourteenth International Conference on Networks (ICN 2015) IARIA, 2015, in press.

[4] Communications Fraud Control Association, "Global Fraud Loss Survey," October 2013. [Online]. Available from: http://www.cfca.org/pdf/survey/CFCA2013GlobalFraudLossSurvey-pressrelease.pdf 2014.06.23.

[5] M. Taniguchi, M. Haft, J. Hollmen, and V. Tresp, "Fraud detection in communication networks using neural and probabilistic methods," in Proceedings of the 1998 IEEE International Conference on: Acoustics, Speech and Signal Processing, vol. 2, 1998, pp. 1241-1244.

[6] P. Burge and J. Shawe-Taylor, "Detecting cellular fraud using adaptive prototypes," in Proceedings AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Management, AAAI Press, 1997, pp. 9-13.

[7] C. S. Hilas and P. A. Mastorocostas, "An application of supervised and unsupervised learning approaches to telecommunications fraud detection," Knowledge-Based Systems, vol. 21, no. 7, pp. 721-726, 2008.

[8] H. Grosser, P. Britos, and R. García-Martínez, "Detecting fraud in mobile telephony using neural networks," in Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence, Bari, Italy, Springer-Verlag, 2005, pp. 613-615.

[9] heise online, "Report: Deutsche Telekom analyzes call data of several calls," [Online]. Available from: http://www.heise.de/newsticker/meldung/Bericht-Deutsche-Telekom-wertet-Verbindungsdaten-saemtlicher-Telefonate-aus-1933436.html 2014.06.23.

[10] AVM GmbH, "Security notice: suspected phone fraud," 06 02 2014. [Online]. Available from: https://www.avm.de/de/News/artikel/2014/sicherheitshinweis_telefonmissbrauch.html 2014.06.23.

[11] R. Eikenberg, "Hack on AVM routers: Fritzbox breach disclosed, millions of routers at risk," 07 03 2014. [Online]. Available from: http://www.heise.de/security/meldung/Hack-gegen-AVM-Router-Fritzbox-Luecke-offengelegt-Millionen-Router-in-Gefahr-2136784.html 2014.04.04.

[12] TELES AG, official homepage. [Online]. Available from: http://www.teles.com/en/teles.html 2014.06.23.

[13] Y. Moreau, H. Verrelst, and J. Vandewalle, "Detection of mobile phone fraud using supervised neural networks: a first prototype," in Proceedings of the 7th International Conference on Artificial Neural Networks, Springer-Verlag, 1997, pp. 1065-1070.

[14] T. Kapourniotis, T. Dagiuklas, G. Polyzos, and P. Alefragkis, "Scam and fraud detection in VoIP networks: analysis and countermeasures using user profiling," in 50th FITCE Congress, 2011, pp. 1-5.

[15] T. Fawcett and F. Provost, "Adaptive fraud detection," Data Mining and Knowledge Discovery, vol. 1, no. 3, pp. 291-316, 1997.

[16] D. Olszewski, J. Kacprzyk, and S. Zadrożny, "Employing Self-Organizing Map for fraud detection," in The 12th International Conference on Artificial Intelligence and Soft Computing (ICAISC 2013), 2013.

[17] J. Hollmén, V. Tresp, and O. Simula, "A Self-Organizing Map for clustering probabilistic models," in Ninth International Conference on Artificial Neural Networks (ICANN), vol. 2, 1999.

[18] R. Alves et al., "Discovering telecom fraud situations through mining anomalous behavior patterns," in Proceedings of the DMBA Workshop on the 12th ACM SIGKDD, 2006.

[19] D. Hoffstadt et al., "A comprehensive framework for detecting and preventing VoIP fraud and misuse," in International Conference on Computing, Networking and Communications (ICNC), 2014, pp. 807-813.

[20] P. Burge, J. Shawe-Taylor, C. Cooke, Y. Moreau, B. Preneel, and C. Stoermann, "Fraud detection and management in mobile telecommunications networks," in European Conference on Security and Detection, 1997, pp. 91-96.

[21] H. Grosser, P. Britos, and R. García-Martínez, "Detecting fraud in mobile telephony using neural networks," in Proceedings of the 18th international conference on Innovations in Applied Artificial Intelligence, Bari, Italy, Springer-Verlag, 2005, pp. 613-615.

[22] M. Taniguchi, M. Haft, J. Hollmen, and V. Tresp, "Fraud detection in communication networks using neural and probabilistic methods," in Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 2, 1998, pp. 1241-1244.

[23] "Der Bundesbeauftragte für den Datenschutz und die Informationsfreiheit: Bundesdatenschutzgesetz (BDSG)," [Online]. Available from: http://www.bfdi.bund.de/SharedDocs/Publikationen/GesetzeVerordnungen/BDSG.pdf?__blob=publicationFile 2014.06.23.

[24] P. Burge and J. Shawe-Taylor, "Detecting cellular fraud using adaptive prototypes," in Proceedings AAAI-97 Workshop on AI Approaches to Fraud Detection and Risk Management, AAAI Press, 1997, pp. 9-13.

[25] N. Grira, M. Crucianu, and N. Boujemaa, "Unsupervised and semi-supervised clustering: a brief survey," in A Review of Machine Learning Techniques for Processing Multimedia Content, Report of the MUSCLE European Network of Excellence (6th Framework Programm), 2005.

[26] WEKA, Machine Learning Group at the University of Waikato, official homepage. [Online] Available from: http://www.cs.waikato.ac.nz/ml/weka/ 2014.12.15.

[27] D. Wang, Q.-y. Wang, S.-y. Zhan, F.-x. Li, and D.-z. Wang, "A feature extraction method for fraud detection in mobile communication networks," in Fifth World Congress on Intelligent Control and Automation, vol. 2, 2004, pp. 1853-1856.

[28] P. Burge, J. Shawe-Taylor, C. Cooke, Y. Moreau, B. Preneel, and C. Stoermann, "Fraud detection and management in mobile telecommunications networks," in European Conference on Security and Detection (ECOS 97), 1997, pp. 91-96.

A Framework Balancing Privacy and Cooperation Incentives in User-Centric Networks

Alessandro Aldini
University of Urbino "Carlo Bo"
Urbino, Italy
email: alessandro.aldini@uniurb.it

Abstract—**User-centricity subsumes new models of Internet connectivity and resource sharing, which are based on collaborative behaviors asking for cooperation strategies. On one hand, typical incentives stimulating cooperation, based, e.g., on trust and remuneration, require some level of information disclosure that can be used to outline the user behavior. On the other hand, disclosing such information can be considered as a privacy breach keeping the users from being involved in certain interactions. In this paper, we present a flexible privacy-preserving mechanism trading privacy for trust-based and cost-based incentives. Firstly, the proposed mechanism is validated theoretically through model checking based analysis. Secondly, implementation issues are discussed with respect to the design of ad-hoc solutions based on a centralized reputation system and a distributed trust system.**

Keywords–user-centric networks; privacy; trust; cooperation incentives; model checking.

I. INTRODUCTION

Nowadays, user-driven services, like personal hotspot and peer-to-peer, play a fundamental role in the reshaping of the Internet value chain. The growing trend towards autonomic user-centric architectures is moving the focus on the user experience, related needs, expectations, and attitude to cooperation. One of the key factors behind the success of community-scale user-centric initiatives is given by the user involvement as a *prosumer*, i.e., an actor combining the roles of service producer and consumer. Such an involvement can be guaranteed only by taking into account several orthogonal aspects, including the need for incentive mechanisms stimulating the willingness to collaborate, the user perception of the trustworthiness of agents and means supporting the community infrastructure, the major issues related to information privacy and risk management arising in a framework favoring the active participation of unknown users.

In a recent work presented at SECURWARE 2014 [1], a novel approach has been proposed to set up a flexible and efficient cooperation infrastructure favoring collaborative behaviors on the basis of specific user's needs in terms of social (e.g., personal sensibility to trust and privacy issues) and economical (e.g., in terms of costs that can be afforded) requirements. The objective of this work – which is a revised and extended version of [1] partially based also on material appeared in [2] – is to show that different dimensions of the problem surveyed above, like trust, privacy, and cooperation incentives, can be effectively balanced to fulfill all the user requirements at the basis of an active involvement as a prosumer in user-centric networks.

The first fundamental aspect governing any interaction in user-centric networks is trust [3]. Establishing stable trustworthiness relations among unknown users is the objective of trust and reputation systems [4]. Trust can be viewed as the subjective belief by which an individual expects a given entity to perform with success some activity on which individual's welfare depends. Reputation emerges implicitly or explicitly in the community as an objective estimation about the level of honesty, integrity, ability, and disposition of each user as perceived by the other members of the community. It is quite natural to rely on trust and reputation information to take decisions about the opportunity to collaborate with certain partners. To this aim, several explicit mechanisms providing estimations of trust and reputation have been proposed in the literature to stimulate and guide cooperation [5], [6], among which we concentrate on those providing computational estimations of user's trustworthiness. Basically, these estimations work effectively as an incentive to collaborate if they represent parameters influencing access to services at favorable conditions, among which we include the service cost as another important aspect affecting the perceived quality of experience. In fact, remuneration is a widely used kind of incentive stimulating cooperation [7], as very often sense of community, synergy, and trust do not suffice to overcome the limitations of obstacles like, e.g., selfishness and, even worse, cheating, which represent threats keeping users from being cooperative. Whenever combined with trust, remuneration enables a virtuous circle for the proliferation of user-centric services.

On the other hand, trust is a concept that may involve and justify the disclosure of personally identifiable sensitive information, which in general can be perceived as a dramatic breach of privacy, thus playing a deterrent role when users are getting involved in interactions. In practice, the lower the attitude to expose sensitive information is, the higher the probability of being untrusted when negotiating a service. Trading privacy for trust is thus a way for balancing the subjective value of what is revealed in exchange of what is obtained [8].

These considerations motivate the need for a flexible cooperation model in the setting of user-centric networks. In the following, we first comment on related work to emphasize the kind of flexibility we would like to obtain with respect

to classical models that integrate trust and privacy. Then, in Section II we describe a novel cooperation model in which privacy is managed and traded with trust. In Section III, we analyze formally the proposed model, even through a comparison with classical ones. This is done in the setting of a real-world cooperation system for user-centric networks [9]. The aim is not only to show that a balanced tradeoff between privacy and trust can be achieved, but also to emphasize the impact of such a tradeoff upon other aspects – like the service cost – that are in some relation with trust. Formal modeling and analysis are based on automata theory and model checking [10]. In Section IV, we discuss all the implementation issues of the proposed approach. In particular, the applicability of the cooperation model is shown under two main practical scenarios. On one hand, we first discuss a centralized reputation-based approach to the implementation of the novel model of privacy management. This framework is based on the presence of a trusted third party (TTP) collecting information about every transaction completed. By combining the subjective estimations on the trust towards each user, the TTP makes them available to the community with the aim of making explicit a collective notion of reputation, while keeping the desired level of privacy for every user involved. On the other hand, we show how to implement the same model in the setting of distributed systems that cannot rely on TTP at run time. In such a case, a trust system is implemented that is based on user's personal experience and, possibly, recommendations provided by neighbors. Finally, some conclusions terminate the paper in Section V.

A. Related Work

Trust and privacy represent two pillars for any social platform aiming at offering resource and information sharing among users [11]. Trading several different cooperation incentives stimulates honest behaviors while keeping users from cheats and selfishness. For instance, it is well-known that making trust and service cost mutual dependent is a winning strategy in the setting of user-centric networks [9], [12], [13], as also proved formally by means of formal methods, like game theory and model checking [14]–[18]. Combining these aspects also with user's privacy is a challenging issue. As an example, the unavoidable contrast between privacy and trust is mitigated by the approach proposed in [19], where it is shown that these two aspects can be traded by employing a mechanism based on *pseudonyms*. In practice, users create freely pseudonyms identified by the so-called *crypto-id*, i.e., the hash of the public key of a locally generated asymmetric cryptography key pair. Then, in different environments, a user can use different pseudonyms to carry out actions logged as events signed with the private key of the chosen pseudonym. If needed to acquire more reputation, several pseudonyms can be linked together in order to augment the number of known actions and potentially increase the trust in the linked entity. However, in approaches such as this one the link is irrevocable.

Developing trust based schemes to enforce trustworthy relations in anonymity networks is another active research field (see, e.g., [20] and the references therein). Incentive mechanisms are proposed in [21] to achieve a balanced tradeoff between privacy and trust in the setting of data-centric ad-hoc networks. In [22], such an interplay is formulated as an optimization problem in which both privacy and trust are expressed as metrics. In [23], trust towards an entity is used to take decisions about the amount of sensitive information to reveal to the entity. Further works on unlinkability [24] and pseudonymity (see, e.g., [25], [26]) provide insights on the tradeoff between privacy and trust.

A typical characteristic of the approaches proposed in the literature is concerned with the incremental nature of privacy disclosure. In fact, sensitive information linking is irrevocable and, as a consequence, any privacy breach is definitive. Instead, the approach proposed in this work aims at relaxing such a condition.

We conclude the state-of-the-art presentation by citing two practical systems that deal with privacy and trust management in cooperative networks. Identity Mixer [27] allows users to control and minimize the amount of personal data they have to reveal in any access request. By selectively disclosing only the information strictly needed for access, different transactions performed by the same user become unlinkable, thus avoiding tracking of users. U-Prove [28] provides a cryptographic platform allowing users to minimally disclose certified information during transactions. In particular, user credentials are generated dynamically and encode only the attributes chosen by the user in a way that makes different transactions unlinkable. In both systems, differently from our proposal, unlinkability is a semantic notion depending on the specific attributes involved in the transactions. Moreover, no computational notion of trust is explicitly employed.

As a specific contribution of our approach that makes it different with respect to other methodologies, a collective notion of trust is employed that ensures privacy through identity obfuscation and offers incentive mechanisms stimulating honest, collaborative behaviors in user-centric networks.

II. Modeling Privacy Management

In a classical view of privacy, a user exposes (part of) personal information in order to be trusted enough to get access to the service of interest. In other words, privacy disclosure is traded for the amount of reputation that the user may need to be considered as a trustworthy partner in some kind of negotiation in which, e.g., service cost may depend on trust. Typically, once different pieces of sensitive information, say I_1 and I_2 (which may represent credentials, virtual identities, or simply the proof of being the user involved in a transaction previously conducted), are linked and exposed to be trusted by someone else, then such a link is irrevocably released. In this view, we say that the disclosure of sensitive information is *incremental* along time.

In order to exemplify, as discussed in [19], I_1 and I_2 may identify two different transactions conducted by the user under two different pseudonyms, each one revealing different personal user data. The user is obviously able to show that both I_1 and I_2 are associated with the same origin and, if such a proof is provided, I_1 and I_2 become irrevocably linked together.

As opposite to the scenario discussed above, we envision an alternative model of privacy release in which the link is not definitive. This is achieved if, for each new transaction conducted by the user, the amount of privacy disclosure is *independent* of the information released in previous interactions. Such a flexibility would allow the user to tune the amount of information to disclose in order to negotiate a transaction at the desired level of privacy without taking care of previous and future interactions. With respect to the example above, we intend that once I_1 and I_2 are linked to complete a given transaction, in a future interaction the same user can decide to break such a connection and expose, e.g., only I_1, with the guarantee that I_2 will be not associated with such an interaction.

In order to make it possible such a revocation mechanism, the idea consists of introducing some form of uncertainty associated with the owners of specific actions. Let us explain how to achieve such a condition by employing the virtual identity framework of [19]. As mentioned above, a virtual identity is represented by the crypto-id. The basic idea of the independent model of privacy release is that trust and transactions are mapped to pieces of the crypto-id, called *chunks*, rather than to the crypto-id as a whole.

Consider, e.g., a typical handshake between Alice, who issues a service request, and Bob, who offers the service. Instead of revealing to be Alice, she accompanies the request with a portion of her crypto-id identified by applying a bitmask to the crypto-id through the bitwise AND operation. Therefore, a chunk is a subset of bits of the crypto-id, of which we know value and position. Amount and position of 1's occurrences in the bitmask are under Alice's control.

The transaction is then identified by the chunk chosen by Alice. Hence, trust values (and related variations due to the feedback following the transaction execution) are not associated with Alice directly, but are related to the chunk of bits extracted from Alice's crypto-id through the chosen bitmask. In general, the same chunk is potentially shared by other crypto-ids belonging to several different users. In other interactions, Alice may select different chunks of her crypto-id. Moreover, she can also spend a set of chunks of her crypto-id in order to exploit a combination of the trust associated with each of these chunks. Thanks to the uncertainty relating chunks and associated owners, every time Alice exposes a chunk to Bob in order to negotiate a transaction, Bob cannot link the current transaction to any of the previous transactions conducted (by Alice or by other users) by using the same chunk or one of its possible subsets or supersets.

Example 1. *For the sake of presentation, consider a 8-bit crypto-id, e.g.,* 10010101*, and calculate the chunk revealing the 2nd and 5th bits of the crypto-id. This is obtained through the following bitwise operation:*

	10010101	*(crypto-id)*
AND	00010010	*(bitmask)*
=	000$\underline{1}$00$\underline{0}$0	*(chunk)*

Notice that the same bitmask identifies the same chunk if applied to the crypto-id 00011100*.*

In the following, we say that a crypto-id K matches a given chunk C if there exists a bitmask that, applied to K via the bitwise AND operation, returns C (in this case, we sometimes say also that C matches K). If two crypto-ids K_1 and K_2 coincide for the bit values identified by a certain bitmask, then they both match the resulting chunk. In other words, whenever the two users identified by K_1 and K_2 use such a chunk, then they are indistinguishable from the viewpoint of the other members of the community. When necessary, we use the extended notation C_B to identify a chunk resulting from the application of bitmask B and the usual vector based notation $C_B[i]$ (resp., $B[i]$) to denote the value of the i-th bit of the chunk (resp., bitmask).

In practice, the chunk sharing principle discussed above represents the basic mechanism enabling the form of identity obfuscation needed by the independent model of privacy release. Strictly speaking, the uncertainty relating chunks and owners is not granted absolutely, as it may happen that a chunk identifies univocally a crypto-id, especially whenever the population is small. However, several solutions can be applied to manage the transient phase during which the community is growing, e.g., by injecting fictitious crypto-ids until the critical mass is reached. Hence, we can safely assume that a deterministic matching between chunk and crypto-id is statistically irrelevant.

An important effect of chunk sharing concerns trust and reputation management. In fact, the trust $t(C)$ towards a chunk C represents a collective notion of the trust towards the set S of users with crypto-id matching C. Hence, the approximation with which $t(C)$ represents the actual trustworthiness of the user employing C in the current transaction depends on the size (and composition) of S. Calculating correct trust estimations by just knowing chunks that can identify several different users is just one of the critical aspects. Another one is concerned with the validation of the chunk exposed by the user in a transaction, who is expected to prove to be a proper owner of the chunk without actually revealing the related crypto-id.

While all these practical issues are discussed in Section IV, in the next section we abstract away from any implementation detail and we ask whether, in general, an independent model of privacy release is worth to be considered with respect to classical, incremental disclosure models. As we will see, an answer to such a question can be provided by applying model checking based formal methods.

III. Formal Verification

In order to estimate the validity of the independent model of privacy release, in this section we propose a comparison with an abstraction of standard approaches in which information linking is irrevocable and privacy disclosure is incremental. Such a comparison is based on the evaluation of metrics that reveal how trading privacy for trust influences access to services and related costs.

For this purpose, we employ quantitative formal methods, thanks to which it is possible to estimate rigorously several properties of the system of interest, prior to implementation. The analysis is supported by the software tool PRISM [10],

[29]–[31], which is a model checker encompassing all the ingredients needed to model and verify our case study. More precisely, through PRISM it is possible to build automatically probabilistic models – like discrete-time Markov chains and Markov decision processes – from state-based formal specifications. The tool supports also modeling of stochastic multi-player games, in which nondeterministic choices are governed by distinct players, thus enabling explicitly the verification of different choice strategies. On the semantic models deriving from formal descriptions, quantitative properties expressed in probabilistic extensions of temporal logics can be verified through model checking techniques.

The comparison is conducted by assuming that the two models of privacy release are applied in the setting of a real-world cooperation system [9], in which users providing services, called *requestees*, and recipients of such services, called *requesters*, are involved in a cooperation process balancing trustworthiness of each participant with access to services and related costs. In the following, we briefly describe the original trust model and its relation with service remuneration [9]. Then, after introducing the modeling and verification assumptions, we discuss the analysis results.

A. Trust Model

Trust is a discrete metric with values ranging in the interval $[0, 50]$, such that $null = 0$, $low = 10$, $med = 25$, and $high = 40$. The trust T_{ij} of user i towards credential j (which can be, e.g., a crypto-id or an entity identity) is modeled abstractly as follows:

$$T_{ij} = \alpha \cdot trust_{ij} + (1 - \alpha) \cdot recs_{ij} \quad (1)$$

Parameter $\alpha \in [0, 1]$ is the risk factor balancing personal experience with recommendations by third parties. The trust metric $trust_{ij}$ is the result of previous direct interactions of i with j. Initially, $trust_{ij}$ is set to the dispositional trust of i, denoted by dt_i. After each positive interaction, $trust_{ij}$ is incremented by a factor v. Parameter $recs_{ij}$ is the average of the trust metrics towards j recommended to i by other users. For each service type, the service trust threshold *st* represents the minimum trust required to negotiate the service.

B. Service Cost Model

The joint combination of trust and remuneration is implemented by making the service cost function dependent on the trust T of the requestee towards the requester credential. The other main parameters are: C_{min}, which is the minimum cost asked by the requestee regardless of trust, C_{max}, which is the maximum cost asked to serve untrusted requests, and the threshold values T' and T'', such that $T'' < T'$.

The cost function proposed in [9] expresses linear dependence between trust and cost:

$$C(T) = \begin{cases} C_{min} + \frac{C_{max} - C_{min}}{T'} \cdot (T' - T) & \text{if } T < T' \\ C_{min} & \text{otherwise} \end{cases} \quad (2)$$

In order to examine thoroughly the trust/cost tradeoff, we consider two more functions approximating the linearity of the relation between trust and cost. In particular, a simple one-step function is as follows:

$$C(T) = \begin{cases} C_{max} & \text{if } T < T' \\ C_{min} & \text{otherwise} \end{cases} \quad (3)$$

while a possible two-steps function is as follows:

$$C(T) = \begin{cases} C_{max} & \text{if } T < T'' \\ C_{max}/2 & \text{if } T'' \leq T < T' \\ C_{min} & \text{otherwise} \end{cases} \quad (4)$$

C. Modeling Assumptions

Our objective is to compare the model of incremental release of privacy (represented in the figures by the curves named inc) with the model of independent release of privacy (represented in the figures by the curves named ind). For the sake of uniformity, for both models we assume abstractly that privacy is released (through the pseudonyms mechanism [19] and through the chunk mechanism, respectively) as a percentage of the total amount of sensitive information that the user may disclose. Similarly, in every trust-based formula we consider percentages of the trust involved.

The experiments are conducted by model checking several configurations of the system against formulas expressed in quantitative extensions of Computation Tree Logic [10]. For instance, Figure 1 refers to one requester interacting with one requestee with the aim of obtaining 10 services that can be of three different types. The figure reports the results for the best strategy, if one exists, allowing the requester to get access to all the services requested by minimizing the total expected cost (reported on the vertical axis) depending on the amount of revealed sensitive information (reported on the horizontal axis). The choice of the amount of privacy to spend for each request is under the control of the requester. The choice of the service type is either governed by the requester, or it is probabilistic with uniform distribution (see the curves denoted by $prob$ in the figure). Requestee's parameters are $dt = med$ and $v = 5$, as we assume that each transaction induces a positive feedback. The three service types are characterized by $st_1 = null$ and (2), $st_2 = low$ and (3), $st_3 = med$ and (4), respectively. The service cost parameters are $C_{min} = 0$, $C_{max} = 10$, $T' = high$, and $T'' = med$.

In order to focus on the difference between the two privacy models whenever the choice of the service is under the control of the requester, we also propose a sensitivity analysis with respect to parameter dt, where we concentrate on the interval of privacy values in which the previous experiment emphasizes the gap between the two models, see Figure 2.

We complete the comparison with an experiment assuming one requester and two requestees, which are chosen nondeterministically by the requester. The number of issued requests is 10, while we consider only the first type of service. The analysis, reported in Figure 3, proposes the results obtained by changing the service cost function. Requestee's trust parameters are as follows: $dt = med$, $st = null$, $\alpha = 0.5$.

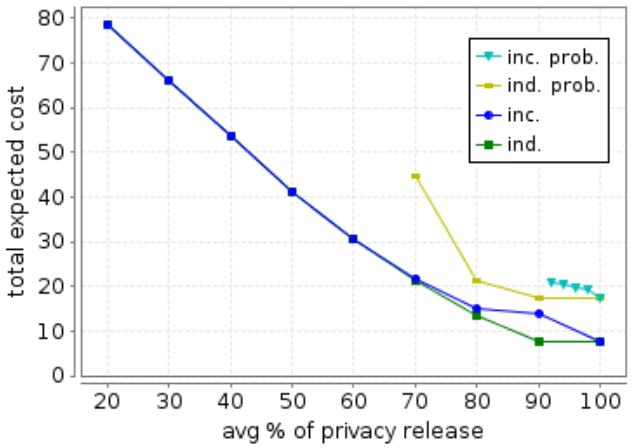
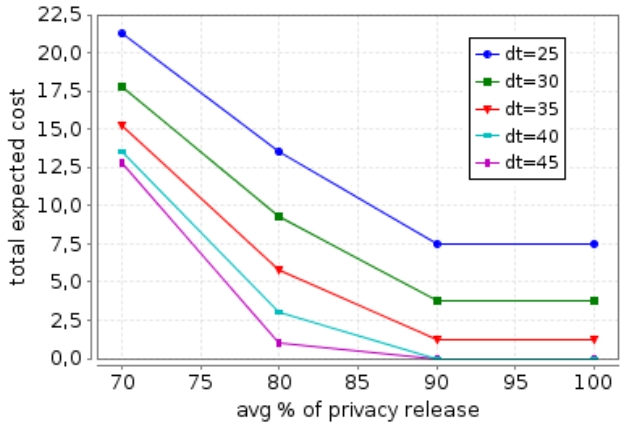
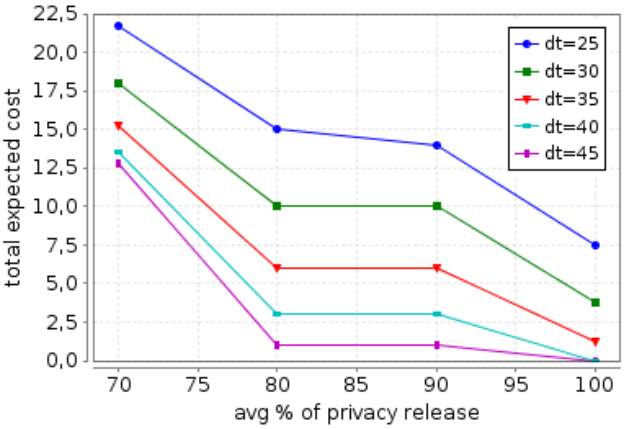


Figure 1. Trading cost for privacy.



(a) Independent privacy release.



(b) Incremental privacy release.

Figure 2. Trading cost for privacy by varying dispositional trust.

D. Evaluation

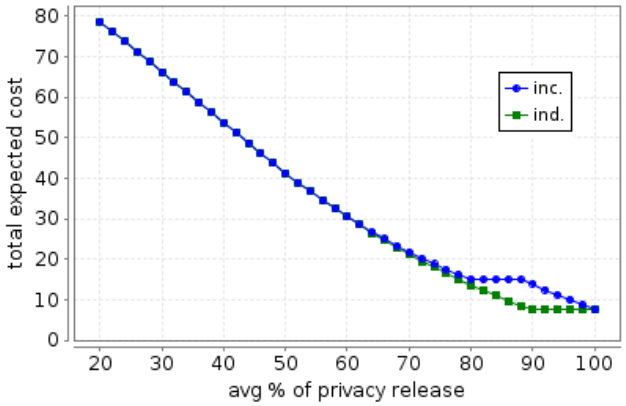
We now comment on the obtained results, by first considering Figure 1, which reveals two interesting behaviors.

Firstly, if the choice of the service is under the control of the requester, then the difference between the two models is significant only for values of the privacy release higher than 70%. In order to interpret this result, we checked the best requester's strategy, which consists of choosing always the service offering the best ratio trust/cost, i.e., the one using (2). Whenever trust is high enough to apply the minimum cost, then it turns out to be convenient to select also the other two service types. According to this strategy, if the privacy disclosure is below 70% it happens that trust does not reach the threshold T'. Therefore, as a consequence of (2), the relation between trust and cost is always linear and the two privacy models turn out to be equivalent from the economic standpoint. On the other hand, if the requester is highly trustworthy, then the cost to pay becomes constantly equal to the minimum cost, meaning that the requester could invest less privacy to obtain the same cost, thus revealing the advantages of the independent model. In practice, independently of the privacy model, it is economically convenient for the requester to disclose the information needed to obtain rapidly the best cost. Instead, for high levels of trust, it would be convenient for requester's privacy to reduce as much as possible the amount of disclosed information. Whenever identity of the requester is always fully disclosed, then the two models experience the same performance.

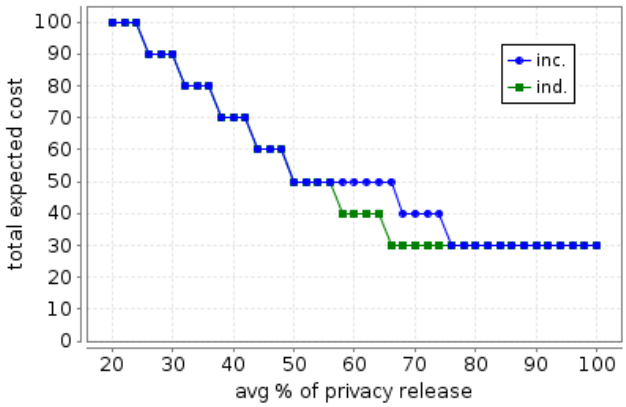
Secondly, if the choice of the service is probabilistic, thus modeling, e.g., a situation in which the requester may require every type of service independently of their cost, then it is not possible to satisfy all the requests if a minimum disclosure of privacy is not guaranteed. However, such a minimum value is considerably higher for the incremental model, in which case at least an average privacy release of 92% is needed. Hence, if the requester is somehow forced to require certain services, then the independent model performs better.

The analysis of Figure 2 confirms the results above also by varying the dispositional trust of the requestee, which is a parameter that does not affect the comparison between the two models. Different results are instead obtained by studying the role of the service cost function, as emphasized by the curves of Figure 3, which show that when step functions are used, the independent model is able to exploit better the intervals of trust in which the service cost is constant.

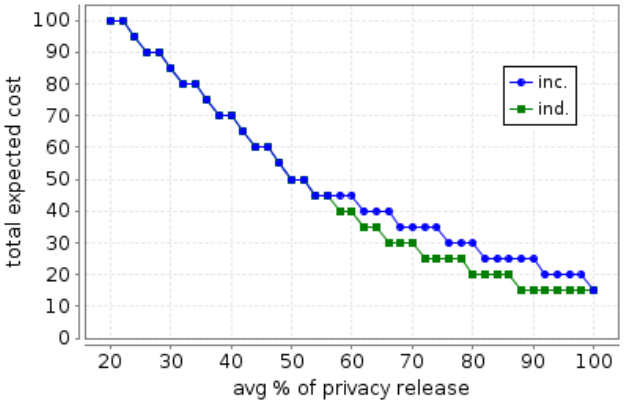
In the previous experiments, priority is given to service cost and to the average disclosure of privacy needed to optimize such a cost. However, if cost is not a fundamental issue, then the tradeoff of interest concerns trust and privacy. In order to analyze such a tradeoff, we reformulate the experiment of Figure 1 by focusing on the optimization of the average percentage of privacy release needed to obtain 10 services of a given type. The results are reported in Table I and refer to the second and third service types, for which the service trust threshold is *low* and *med*, respectively. Since to obtain such services the requester must be trusted by the requestee, we examine the tradeoff between such a trust and requester's privacy. For each of the two cases, the observed values show that through the independent model we obtain all the required services by disclosing much less privacy



(a) Cost Equation 2.



(b) Cost Equation 3.



(c) Cost Equation 4.

Figure 3. Trading cost for privacy by varying cost function.

than through the incremental model. The related difference is directly proportional to the trust threshold needed to obtain the services.

IV. Implementation Issues

The independent model of privacy release is based on the notion of virtual identity represented by means of the crypto-id, which we assume to be calculated using a cryptographic hash function, like SHA-3, over the public key of an asymmetric cryptography key pair generated by the user (see, e.g., [32] for a survey on cryptographic primitives). As a notation, we assume that (pk_u, sk_u) is the asymmetric crypto key pair associated with user u, such that pk_u is publicly available and $hash(pk_u)$ represents the related crypto-id.

TABLE I. Trading trust for privacy: avg % of privacy release.

service	inc.	ind.
type 2 (st_2 = *low*)	38%	28%
type 3 (st_3 = *med*)	92%	64%

Whenever issuing a service request, Alice chooses a bitmask that is applied to her crypto-id in order to extract the chunk according to the mechanism explained in Section II. Then, Alice sends to Bob a ciphertext (generated using Bob's public key) containing the chunk and a cryptographic proof for the request demonstrating that the chunk exposed is actually extracted from the crypto-id of the user issuing the request. In the following, we propose two solutions for the generation of such a proof that preserve anonymity of Alice's crypto-id.

In a centralized scenario, we assume that crypto-ids are stored in a non-public repository managed by a trusted, central authority (CA). In this case, the cryptographic proof may consist of a blind signature [33] obtained by Alice from the CA prior using the chunk, as explained in the following and shown in Figure 4. In the proposed protocol, (e_A, d_A) denotes an asymmetric crypto key pair generated by Alice to implement the blind signature.

Before issuing a service request associated with chunk C, Alice signs (with her private key sk_A) a request to the CA containing C and the encryption (using e_A) of the hash of a timestamp, $(C, (hash(t))_{e_A})_{sk_A}$. The request is accompanied by Alice's identity.

Upon reception of the validation request from Alice, the CA extracts C (using pk_A) and checks its conformity through the public key of Alice, by comparing C against the crypto-id $hash(pk_A)$. Then, if such a check is successful, the CA generates a blind signature including C and the encrypted hashed timestamp, $(C, (hash(t))_{e_A})_{sk_{CA}}$. Notice that the CA can neither guess the timestamp t, nor associate its hashed value with Alice.

When receiving such a ciphertext, Alice strips away her encryption using d_A, thus leaving her with the CA signature of C and of the hash of the timestamp, $(C, (hash(t)))_{sk_{CA}}$.

The request sent to Bob includes C, the timestamp, and the CA signature, $(C, t, (C, (hash(t)))_{sk_{CA}})$.

Hence, Bob can check the signature for validity, by comparing C and t against the content of the message signed by the CA, and then forward the obtained information to the CA, which verifies timestamp doublespending. The reason for using a timestamp is to avoid unauthorized users employing chunks signed by the CA, while the use of the blind signature ensures Alice anonymity. Indeed, notice that the knowledge of C does not allow neither Bob nor the CA to infer the originating crypto-id and, therefore, the identity of Alice. More

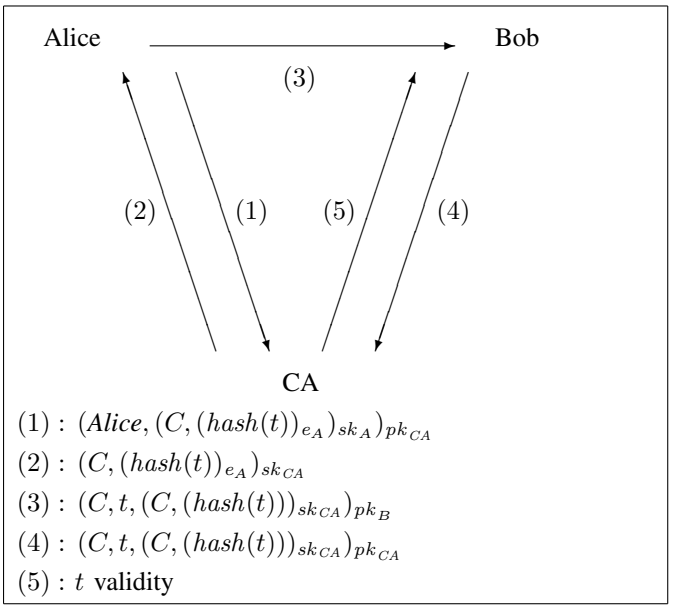


Figure 4. Cryptographic proof of chunk C through the CA.

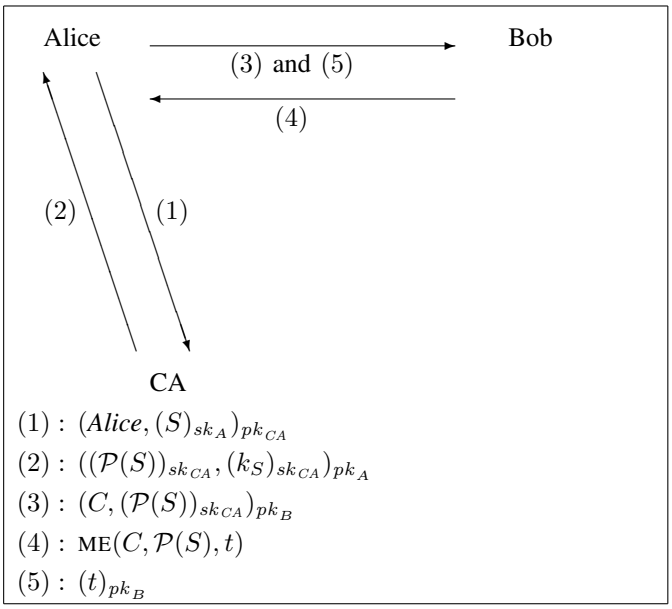


Figure 5. Cryptographic proof of chunk C in zero-knowledge.

sophisticated blind signature schemes can be used, e.g., to offer fairness [34], in order to revoke blindness in case of suspicious behaviors by some chunk and, therefore, isolate dishonest users.

Alternatively, zero-knowledge proofs can be applied on-the-fly whenever we consider a distributed scenario that does not involve communication with the CA during the transaction lifetime. For instance, zero-knowledge sets and set membership [35], [36] are proposed to decide the membership problem $x \in S$ by preserving as much privacy as possible about S (or x). In particular, a zero-knowledge membership proof works as follows. Let $\mathcal{P}(S)$ be a privacy-preserving token of a set S (e.g., a certified commitment by the CA on the set S that does not reveal any information about its constituting elements) and x an element belonging to S. Whenever the verifier knows the pair $(\mathcal{P}(S), x)$, the prover can convince the verifier in zero-knowledge that $x \in S$ without leaking anything about S to the verifier. Membership encryption [37] is a cryptographic technique extending membership proof in which:

- $\mathcal{P}(S)$ is generated from S and a secret key k_S;
- the encryption algorithm, called ME, takes as input x, $\mathcal{P}(S)$, and the message m to encrypt;
- the decryption algorithm, called MD, requires the pair (S, k_S) and holding the membership $x \in S$ to return successfully m.

In our setting, x is represented by the chunk C used by Alice, while S is given by the set of chunks committed by the CA whenever Alice registers her crypto-id in the CA repository. Hence, Bob plays the role of verifier whenever Alice exposes chunk C and the certified token $\mathcal{P}(S)$. The handshake works as illustrated in Figure 5.

Initially, Alice signs a request to the CA including the list S of chunks she intends to use in future interactions (such a request can be renewed if Alice requires more chunks).

The CA verifies whether S is correct with respect to Alice's crypto-id, i.e., each of its elements matches the hash of the public key of Alice. If this is the case, the CA generates the privacy-preserving token $\mathcal{P}(S)$ and the secret key k_S, and then signs such a pair for Alice.

Afterwards, when issuing a request to Bob, Alice sends the chosen chunk C, which is expected to belong to S, and the certified token $\mathcal{P}(S)$.

Bob calculates $\text{ME}(C, \mathcal{P}(S), t)$, where t is a timestamp chosen by Bob, and then asks Alice to extract successfully t to prove that $C \in S$.

Finally, Alice computes $\text{MD}(\text{ME}(C, \mathcal{P}(S), t), S, k_S)$, which is equal to t if and only if $C \in S$. It is worth noticing that the membership proof from membership encryption is non-transferable, i.e., Bob cannot convince any third party that $C \in S$, thus ensuring the privacy of $\mathcal{P}(S)$.

Once Bob accepts a request accompanied by chunk C, he must estimate trustworthiness towards C in order to negotiate the service parameters. In the following, we propose a centralized reputation based approach and a distributed trust based approach. To this aim, we assume to deal with a numeric, totally ordered domain $\mathcal{T}$ for trust and reputation values and that the evaluation feedback at the end of every transaction is reported as a positive/negative variation.

A. Design of a Centralized Reputation System

The key feature of the proposed approach to privacy management is that any transaction is associated with portions, called chunks, of the crypto-id representing the virtual identity. The same chunk can be shared by different users, who decide for each transaction the chunk size and whether to combine

together chunks previously used. Hence, the relation among chunks and related crypto-ids must be managed carefully in order to estimate correctly the reputation of users.

For this purpose, the centralized reputation system we propose is managed by the CA, which is in charge of two main tasks:

1) management of the reputation of each crypto-id on the basis of the feedback reported about the chunks that match the crypto-id;
2) calculation of the reputation of the chunk spent in a transaction on the basis of the reputations of the crypto-ids matching the chunk.

Since in a limiting scenario a chunk could be a single bit, based on such a granularity we assume that reputation is managed at the bit level.

As far as the first task of the CA is concerned, whenever at the end of a transaction a user transmits the feedback concerning a chunk C, the CA is not able to infer from which crypto-id C is actually originated. Hence, the CA distributes the result v of the user evaluation among the bits of C for every crypto-id matching C. More precisely, the bit reputation variation is $\delta \cdot v$, where δ is a discounting factor in $[0, 1]$ proportional to the size of the chunk. On one hand, the role of δ is to strengthen the relation between the amount of sensitive information exposed by the user in a transaction and the trustworthiness towards such a user. On the other hand, δ mitigates the effect of the use of small chunks, as they are shared by a larger number of users and, therefore, they represent very roughly the users employing them.

Example 2. *Consider four users with the following crypto-ids:*

$$K_1 : 10010100 \quad K_2 : 00010010$$
$$K_3 : 01110111 \quad K_4 : 11011011$$

and an initial situation in which the vector of bit's reputation is $rep_{K_i} = 00000000$*, for* $1 \leq i \leq 4$*. If user* 1 *employs bitmask* 01110000 *for a transaction evaluated positively with* $v = 1$ *(and* $\delta = 1$*), then, the update performed by the CA is* $rep_{K_1} = 01110000$ *and* $rep_{K_2} = 01110000$*, because the chunk used is shared by users* 1 *and* 2*.*

Then, if user 3 *uses bitmask* 00011100 *and the feedback is as above, we obtain the reputation changes* $rep_{K_1} = 01121100$ *and* $rep_{K_3} = 00011100$*.*

Finally, if user 4 *uses bitmask* 00000111*, then any feedback is applied to (the first three bits of)* K_4 *only.*

As shown by the example, the choice of the chunk does not ensure perfect privacy of the user with respect to the CA. Similarly as discussed in Section II, the probability of an unequivocal identification of the crypto-id by the CA depends on the size of the community and of the chunk.

As far as the second task of the CA is concerned, the calculation of the reputation of a chunk deals with the same issues surveyed above. Whenever a user forwards to the CA a chunk C in order to know the related reputation, the CA could not be able to infer the identity of the originating crypto-id. Thus, the reputation of C results from a combination (through the arithmetic mean) of the reputations of such a chunk within every crypto-id K matching C.

Let $rep_K(C)$ be the reputation of chunk C within the crypto-id K, which is calculated by summing up the reputations of the bits of K forming C. By default, $rep_K(C) = 0$ if K does not match C. Moreover, let $\mathcal{M}(C)$ denote the number of crypto-ids matching C. Then, the reputation of chunk C is:

$$\frac{1}{\mathcal{M}(C)} \cdot \sum_K rep_K(C) \tag{5}$$

where the summation is over all the crypto-ids K registered in the CA repository.

Example 3. *With reference to the previous example, consider a new transaction in which user* 1 *employs the bitmask* 01110000*. The resulting chunk is shared by users* 1 *and* 2*. Hence, by using (5), its reputation is:*

$$\tfrac{1}{2} \cdot ((1 + 1 + 2) + (1 + 1 + 1)) = 3.5$$

B. Design of a Distributed Trust System

Handling trust towards users by tracing the usage of (possibly shared) chunks is a hard task in the absence of a centralized reputation system. To deal with this problem, in order to estimate user's trustworthiness we define a local trust structure that allows any user offering a service to associate a trust value with every chunk received to negotiate the service. In particular, the proposed approach does not rely on the knowledge of the list of crypto-ids.

Let $\mathcal{C}$ be the set of chunks with which the user has interacted in completed transactions. The local trust structure is based on the definition of a partially ordered set (*poset*, for short) $(\mathcal{C}, \leq)$ over set $\mathcal{C}$ with respect to a partial order $\leq$. We recall that to define a poset, the binary relation $\leq$ must be reflexive, antisymmetric, and transitive for the elements of $\mathcal{C}$. In the rest of the section, we call $\leq$ refinement operator, which is defined as follows.

Definition 1 (Chunk refinement)**.** *Let* n *be the crypto-id size. Given chunks* $C_B, C_{B'}$*, we say that* $C_{B'}$ *refines* C_B*, denoted* $C_B \leq C_{B'}$*, if and only if:*

- *for all* $1 \leq i \leq n$*:* $B[i] \leq B'[i]$*;*
- *for all* $1 \leq i \leq n$*: if* $B[i] = 1$ *then* $C_B[i] = C_{B'}[i]$*.*

Notice that if $C_B \leq C_{B'}$ then B is a submask of B' and the information exposed by $C_{B'}$ includes that revealed by C_B. The intuition is that if two chunks are related through $\leq$ then they could be originated from the same crypto-id.

As we will see, maintaining the poset structure provides the means to approximate the trust towards any crypto-id by employing the trust related to the potential constituting chunks. Each element of the poset $(\mathcal{C}, \leq)$ is labeled by a value of the trust domain $\mathcal{T}$. Such a value represents the trust of the user towards the related chunk resulting from interactions associated with such a chunk. Formally, we denote such an extended structure with $(\mathcal{C}, \leq, t)$, where $t : \mathcal{C} \to \mathcal{T}$ defines the mapping from chunks to trust values. Initially, for every unknown chunk C with which the user interacts for the first time, we assume $t(C)$ to be equal to the dispositional trust dt of the user, which represents the attitude to cooperate with unknown users.

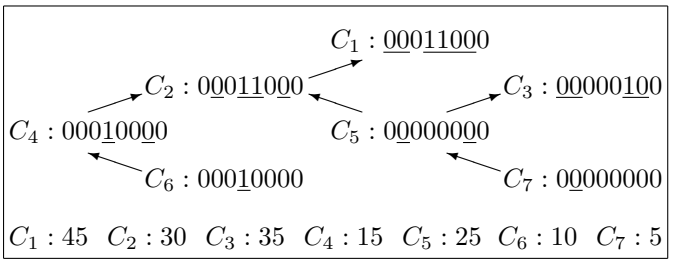


Figure 6. Example of a local trust structure.

Example 4. *Figure 6, which in the following we use as running example, shows the graphical representation of a poset, where, e.g., $C_6 \leq C_4 \leq C_2 \leq C_1$, as well as $C_7 \leq C_5 \leq C_3$, while, e.g., C_6 and C_3 are not related with each other. Moreover, the figure reports also the trust associated with each known chunk at a given instant of time, by assuming the trust domain* $[0, 50]$.

To emphasize the nature of the independent model of privacy release, notice that even if Alice invested chunk C_1 in a past interaction with Bob, whose reference trust structure is that depicted in Figure 6, then in the current transaction she may use chunk C_2 only, while Bob cannot infer the link between the user of the past interaction associated with C_1 and the current one. As a side effect, notice also that all the users with a crypto-id matching C_2 actually benefit from the trust (or pay the mistrust) associated with C_2.

The obfuscation mechanism illustrated in the example above respects the requirements of the independent model of privacy release discussed in Section II.

Similarly as done in the previous section, we now illustrate how to manage the trust $t(C)$ towards the chunk C on the basis of the feedback v following any transaction associated with C. In particular, the trust variation applied to $t(C)$ is simply $\delta \cdot v$, where δ is the discounting factor discussed in the previous section.

Example 5. *As a consequence of a positive transaction conducted through chunk C_2 and resulting in a trust variation equal to, e.g., $+5$, we would obtain $t(C_2) = 32.5$ if $\delta = 0.5$, and $t(C_2) = 35$ if $\delta = 1$. Notice that in the former case the discounting factor represents the ratio between the size of the chunk used and the size of the originating crypto-id, thus emphasizing that trust is proportional to the amount of information disclosure.*

Once $t(C)$ has been updated by applying the variation $\delta \cdot v$, it is worth deciding whether and how the feedback related to chunk C has to be propagated to other elements of the trust structure $(\mathcal{C}, \leq, t)$. First of all, propagation would result in ambiguity if applied to chunks of the poset that cannot be related through $\leq$, because unrelated chunks cannot be brought back to the same crypto-id. Therefore, the remaining cases refer to the chunks that refine (or are refined by) C.

Depending on the feedback, which can be either positive or negative, the potential application of a discounting factor, and the propagation direction (towards finer or coarser chunks, or else both), every possible combination gives rise to a different propagation policy. Tuning these parameters is a task of the user depending on her/his attitude to cooperation. In the following, we describe a policy balancing accuracy of the trust estimations with robustness against malicious behaviors.

On one hand, negative trust variations are not propagated to elements that refine C, because an interaction disclosing a small amount of sensitive information should not compromise the trust level of chunks that expose more information. The objective of this rule is to contrast potential attacks by users preserving their identity and aiming at penalizing the trust of small chunks shared by a large number of users. On the other hand, in order to overcome the problem of trust underestimation and to fully exploit the flexibility of the independent model of privacy release, positive trust variations are propagated to chunks refining C, while positive/negative trust variations are propagated to every chunk in the poset that is refined by C. Another objective of this rule is to favor, in terms of trust, the disclosure of information. In order to keep under control the propagation mechanism, the trust variation for any chunk C' inherited by the feedback related to chunk C is further discounted by a factor δ' proportional to the difference between the size of C and the size of C'. In practice, the larger the difference between C and C' is, the slighter the impact of the trust variation of C upon C'.

Example 6. *Consider chunk C_2 and the positive transaction of the previous example determining $t(C_2) = 32.5$ (i.e., $\delta \cdot v = 2.5$). Then, by virtue of the propagation policy discussed above we have, e.g., $t(C_4) = 16.25$ and $t(C_6) = 10.625$. Chunk C_5 (resp. C_7) gains the same variation applied to C_4 (resp., C_6). On the other hand, C_1 inherits a discounted trust gain equal to $2.5 \cdot \frac{2}{3}$, because C_1 refines C_2 and the trust variation is positive, while C_3 does not inherit any trust gain, because C_2 and C_3 are not related with each other.*

The local trust structure continuously evolves not only by virtue of the updates discussed above, but also as a consequence of the treatment of new chunks. Associating a new chunk C that is added to the poset with the dispositional trust of the user is a policy that does not take into account the knowledge of the trust structure $(\mathcal{C}, \leq, t)$, which can be employed to infer some trust information about C.

Based on the same intuition behind feedback propagation, the trust values associated with known chunks that are in some relation with C can be combined to set up the initial value of $t(C)$. In fact, C can be interpreted as an approximation of such chunks. As in the case of the propagation policy, we can envision several different rules, among which we advocate the following one: $t(C)$ is assigned the arithmetic mean of the trust values associated with chunks that refine C, while those refined by C are ignored. In fact, the accuracy of the trust estimations is directly proportional to the size of the chunks. Therefore, estimating $t(C)$ based on small chunks refined by C would lead to a rough approximation of the trust towards the users employing C. Moreover, the chunks refining C that are considered must be pairwise unrelated by $\leq$ in order to avoid redundancy when counting the related trust values.

Definition 2 (Chunk coverage)**.** *Let $(\mathcal{C}, \leq, t)$ be a trust structure and $C \notin \mathcal{C}$ a new chunk that must be added to the poset. A coverage for C is a set $\mathcal{K} = \{C_1, \ldots C_m\} \subseteq \mathcal{C}$ such that:*

- *$C_i \not\leq C_j$ for all $1 \leq i, j \leq m$;*
- *$C \leq C_i$ for all $1 \leq i \leq m$.*

The initial value of $t(C)$ induced by the coverage $\mathcal{K}$ is:

$$t_{\mathcal{K}}(C) = \frac{1}{m} \cdot \sum_{i=1}^{m} t(C_i).$$

It is worth noticing that the poset may enable several different coverages for a chunk C. If $\mathcal{K}_1, \ldots, \mathcal{K}_p$ are the possible coverages for C in the trust structure $(\mathcal{C}, \leq, t)$, then whenever C is added to $\mathcal{C}$ we set:

$$t(C) = f\{\!|\, t_{\mathcal{K}_i}(C) \mid 1 \leq i \leq p \,|\!\}$$

where f is an associative and commutative arithmetical function (like, e.g., min, max, and avg) applied to the multiset of initial trust values induced by the different coverages.

Example 7. *Consider the trust structure of Figure 6. A coverage for chunk C_8 : 00000000 is the set $\mathcal{K} = \{C_4, C_5\}$, which induces the initial trust value $t_{\mathcal{K}}(C_8) = 20$. Other candidates are $\{C_2, C_3\}$, $\{C_3, C_4\}$, and $\{C_1\}$. Therefore, the initial trust resulting from the application of function avg is 30.625, while we obtain 45 for function max and 20 for function min.*

In general, from the effectiveness standpoint, the trust structure $(\mathcal{C}, \leq, t)$ is used to manage locally information (about chunk's trust) allowing the user to approximate the trust towards other users, without any knowledge about their crypto-ids and actual behaviors. As far as efficiency issues are concerned, in order to circumvent the problem of dealing with a huge trust structure, it is possible to constrain a priori the number of different chunks that can be chosen by every user.

C. Evaluation

The chunk based identity sharing mechanism of the independent model of privacy release has several impacts upon the functionalities of the reputation and trust systems. As a consequence of chunk sharing, the crypto-ids matching the same chunk actually benefit from the reputation (or pay the mistrust) associated with such a chunk. This aspect is crucial for the requirements of the independent model of privacy release and can be viewed as an incentive to take honest decisions, because a high number of trustworthy chunks contribute to increase the probability of obtaining services at a reasonable cost by preserving the desired level of privacy. Hence, all the users sharing trustworthy chunks benefit from this virtuous circle.

Obviously, it is beneficial for a chunk C if all users controlling it are trustworthy. However, if at least one of them is very untrustworthy and carries out at least one illegal action linked to C, then the chunk may rapidly become untrustworthy and useless for all other users. In addition, if C becomes untrustworthy then it may also impact the trustworthiness of any chunk C' such that C and C' match the crypto-id of the same user. These side effects are mitigated implicitly by using mixed cooperation strategies based on trust and cost [14], [16], [17] and explicitly by applying the discounting factors discussed in the previous sections. An effective but severe solution consists of resorting to a CA capable of revoking blindness in case of suspicious behaviors by some chunk, in order to isolate dishonest users and repair the reputation of the chunk involved.

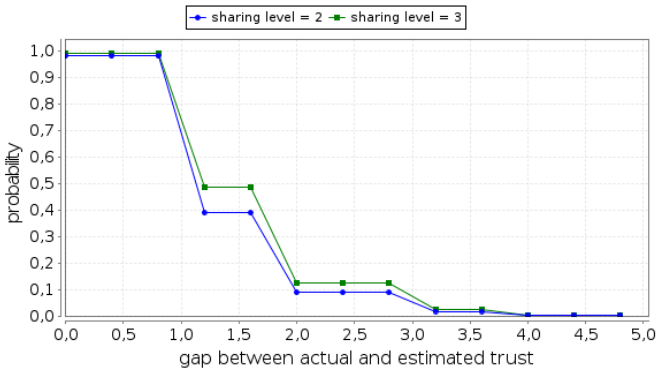


Figure 7. Approximating user trust through chunk trust.

The choice of the chunk size represents another important aspect. In fact, a tradeoff exists between chunk size, privacy, and trust/reputation. The user privileging privacy employs chunks of small size. With high probability, small chunks cannot be used to negotiate favorable service conditions and provide also a rough approximation of the real trustworthiness of the user, as they are shared by a high number of users influencing their usage. On the other hand, the user privileging accuracy of trustworthiness employs chunks of large size, thus sacrificing more privacy as the probability of identification becomes higher.

The discussion above emphasizes that the use of chunks (and of trust information based on them) implies an approximation of the estimation of the trust towards users. In order to quantify the approximation level, an experiment has been conducted by employing the formal framework illustrated in Section II. By assuming a scenario with 4 users, each one having 2 chunks to issue 25 total requests to a service provider, we have evaluated the difference between the *estimated* trustworthiness of each user (as resulting from the combination of the trust of each chunk matching the user crypto-id) and the *actual* trustworthiness of each user (that derives by tracing the actual behavior of the user). For each service request, uniform probability distributions have been used to govern the choice of: the user negotiating the transaction, the chunk exposed, and the feedback reported about the user behavior (no discounting factor is applied). Moreover, we recall that the trust domain is the interval $[0, 50]$.

For this scenario, the curves of Figure 7 evaluate the probability with which the (absolute value of the) difference between actual and estimated trust is higher than the values reported in the horizontal axis. Each curve refers to a different sharing level, which expresses the minimum number of users sharing every chunk. For instance, we observe that the probability that

the trust gap is greater than 2 for a sharing level equal to 2 (resp., 3) is less than 10% (resp., 12%) and rapidly converges to zero.

Finally, the combination of the trust and reputation systems surveyed above can be easily achieved by merging the resulting metrics through the following formula:

$$\alpha \cdot trust(C) + (1 - \alpha) \cdot rep(C) \quad (6)$$

where C is the chunk under evaluation, α is the risk factor, function $trust$ returns the trust resulting from the distributed trust system, and function rep returns either the reputation provided by the CA if a centralized reputation system is available, or a combination (through the arithmetic mean) of the trust values possibly recommended by neighbors. Moreover, we emphasize that the presentation of the proposed design models abstracts away from the specific trust and reputation metrics that are adopted. Indeed, basically, our method may be integrated with any computational notion of trust and with any recommendation mechanism used in classical trust/reputation systems, see, e.g., [38]–[40].

V. Conclusion

The attitude to cooperation is strongly affected by the tradeoff existing among privacy and trustworthiness of the involved parties and cost of the exchanged services. The proposed model of privacy release offers a high level of flexibility in the management of such a tradeoff. In particular, by virtue of a mechanism based on the splitting of crypto-ids, it is possible to manage the disclosure of sensitive information in a less restrictive way with respect to classical models.

To summarize the results obtained from the formal verification, we observe that the major freedom degree of the independent model ensures better performance with respect to the incremental model. This is always true if the main objective is trading privacy for trust. If services must be paid and cost depends on trust, then the adopted cost function affects the tradeoff among privacy, trust, and cost, by revealing the advantages of the independent model in the intervals of trust values in which cost is constant.

From the implementation viewpoint, it has been shown that the novel model can be effectively applied both in centralized reputation systems and in distributed trust systems. The empirical analysis of the peculiarities of each solution, like the bottleneck problem induced by the CA or the efficiency and accuracy ensured by the local trust structure, represents work in progress.

We conclude by observing that a successful deployment of the proposed approach is strictly related to the choice of the trust policies and configuration parameters, which are currently subject to sensitive analysis through formal verification. Solutions to manage the dynamic variability at run time of these parameters are left as future work. Similarly, the approximation induced by the analysis of chunk trustworthiness whenever estimating the actual behavior of users shall be verified in a real-world scenario characterized by a sufficiently large population.

References

[1] A. Aldini, "Saving privacy in trust-based user-centric distributed systems," in 8th Int. Conf. on Emerging Security Information, Systems and Technologies (SECURWARE2014). IARIA, 2014, pp. 76–81.

[2] A. Aldini, A. Bogliolo, C. Ballester, and J.-M. Seigneur, "On the tradeoff among trust, privacy, and cost in incentive-based networks," in 8th IFIP WG 11.11 Int. Conf. on Trust Management, ser. IFIP AICT, J. Zhou et al., Eds., vol. 430. Springer, 2014, pp. 205–212.

[3] A. Aldini and A. Bogliolo, Eds., User-Centric Networking – Future Perspectives, ser. Lecture Notes in Social Networks. Springer, 2014.

[4] A. Jøsang, "Trust and reputation systems," in Foundations of Security Analysis and Design IV (FOSAD'07), ser. LNCS, A. Aldini and R. Gorrieri, Eds. Springer, 2007, vol. 4677, pp. 209–245.

[5] J. Sabater and C. Sierra, "Review on computational trust and reputation models," Artificial Intelligence Review, vol. 24, 2005, pp. 33–60.

[6] E. Chang, F. Hussain, and T. Dillon, Trust and Reputation for Service-Oriented Environments: Technologies For Building Business Intelligence And Consumer Confidence. Wiley, 2005.

[7] S. Greengard, "Social games, virtual goods," Communications of the ACM, vol. 54, no. 4, 2011, pp. 19–22.

[8] S. Taddei and B. Contena, "Privacy, trust and control: Which relationships with online self-disclosure?" Computers in Human Behavior, vol. 29, no. 3, 2013, pp. 821–826.

[9] A. Bogliolo, P. Polidori, A. Aldini, W. Moreira, P. Mendes, M. Yildiz, C. Ballester, and J.-M. Seigneur, "Virtual currency and reputation-based cooperation incentives in user-centric networks," in 8th Int. Wireless Communications and Mobile Computing Conf. (IWCMC'12). IEEE, 2012, pp. 895–900.

[10] V. Forejt, M. Kwiatkowska, G. Norman, and D. Parker, "Automated verification techniques for probabilistic systems," in Formal Methods for Eternal Networked Software Systems, ser. LNCS, M. Bernardo and V. Issarny, Eds. Springer, 2011, vol. 6659, pp. 53–113.

[11] L. Caviglione, M. Coccoli, and A. Merlo, Eds., Social Network Engineering for Secure Web Data and Services. IGI Global, 2013.

[12] Y. Zhang, L. Lin, and J. Huai, "Balancing trust and incentive in peer-to-peer collaborative system," Journal of Network Security, vol. 5, 2007, pp. 73–81.

[13] M. Yildiz, M.-A. Khan, F. Sivrikaya, and S. Albayrak, "Cooperation incentives based load balancing in UCN: a probabilistic approach," in Global Communications Conf. (GLOBECOM'12). IEEE, 2012, pp. 2746–2752.

[14] Z. Li and H. Shen, "Game-theoretic analysis of cooperation incentives strategies in mobile ad hoc networks," Transactions on Mobile Computing, vol. 11, no. 8, 2012, pp. 1287–1303.

[15] A. Aldini and A. Bogliolo, "Model checking of trust-based user-centric cooperative networks," in 4th Int. Conf. on Advances in Future Internet (AFIN2012). IARIA, 2012, pp. 32–41.

[16] A. Aldini, "Formal approach to design and automatic verification of cooperation-based networks," Journal On Advances in Internet Technology, vol. 6, 2013, pp. 42–56.

[17] M. Kwiatkowska, D. Parker, and A. Simaitis, "Strategic analysis of trust models for user-centric networks," in Int. Workshop on Strategic Reasoning (SR'13), vol. 112. EPTCS, 2013, pp. 53–60.

[18] A. Aldini and A. Bogliolo, "Modeling and verification of cooperation incentive mechanisms in user-centric wireless communications," in Security, Privacy, Trust, and Resource Management in Mobile and Wireless Communications, D. Rawat, B. Bista, and G. Yan, Eds. IGI Global, 2014, pp. 432–461.

[19] J.-M. Seigneur and C.-D. Jensen, "Trading privacy for trust," in 2nd Int. Conf. on Trust Management (iTrust'04), ser. LNCS, vol. 2995. Springer, 2004, pp. 93–107.

[20] V. Sassone, S. Hamadou, and M. Yang, "Trust in anonymity networks," in Conf. on Concurrency Theory (CONCUR'10), ser. LNCS, vol. 6269. Springer, 2010, pp. 48–70.

[21] M. Raya, R. Shokri, and J.-P. Hubaux, "On the tradeoff between trust and privacy in wireless ad hoc networks," in 3rd ACM Conf. on Wireless Network Security (WiSec'10), 2010, pp. 75–80.

[22] L. Lilien and B. Bhargava, "Privacy and trust in online interactions," in Online Consumer Protection: Theories of Human Relativism. IGI Global, 2009, pp. 85–122.

[23] W. Wagealla, M. Carbone, C. English, S. Terzis, and P. Nixon, "A formal model of trust lifecycle management," in Workshop on Formal Aspects of Security and Trust (FAST'03), 2003.

[24] S. Köpsell and S. Steinbrecher, "Modeling unlinkability," in 3rd Workshop on Privacy Enhancing Technologies, ser. LNCS, vol. 2760. Springer, 2003, pp. 32–47.

[25] I. Goldberg, "A pseudonymous communications infrastructure for the internet," Ph.D. dissertation, University of California at Berkeley, 2000.

[26] A. Kobsa and J. Schreck, "Privacy through pseudonymity in user-adaptive systems," ACM Transactions on Internet Technology, vol. 3, no. 2, 2003, pp. 149–183.

[27] "Identity Mixer," accessed: 2015-05-15. [Online]. Available: http://www.zurich.ibm.com/idemix/

[28] C. Paquin and G. Zaverucha, "U-Prove cryptographic specification v1.1 (revision 3)," 2013, accessed: 2015-05-15. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=166969

[29] T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis, "Prism-games: a model checker for stochastic multi-player games," in 19th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'13), ser. LNCS, vol. 7795. Springer, 2013, pp. 185–191.

[30] ——, "Automatic verification of competitive stochastic systems," Formal Methods in System Design, vol. 43, no. 1, 2013, pp. 61–92.

[31] M. Kwiatkowska, G. Norman, and D. Parker, "Prism 4.0: verification of probabilistic real-time systems," in 23rd Int. Conf. on Computer Aided Verification (CAV'11), ser. LNCS, vol. 6806. Springer, 2011, pp. 585–591.

[32] J. Katz and Y. Lindell, Introduction to Modern Cryptography – 2nd Edition. CRC Press, 2014.

[33] D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash," in Advances in Cryptology (CRYPTO'88), ser. LNCS, vol. 403. Springer, 1990, pp. 319–327.

[34] M. Stadler, J.-M. Piveteau, and J. Camenisch, "Fair blind signatures," in Eurocrypt'95, ser. LNCS, vol. 921. Springer, 1995, pp. 209–219.

[35] S. Micali, M. Rabin, and J. Kilian, "Zero-knowledge sets," in 44th Symposium on Foundations of Computer Science (FOCS2003). IEEE, 2003, pp. 80–91.

[36] J. Camenisch, R. Chaabouni, and A. Shelat, "Efficient protocols for set membership and range proofs," in 14th Int. Conf. on the Theory and Application of Cryptology and Information Security (ASIACRYPT'08), ser. LNCS, vol. 5350. Springer, 2008, pp. 234–252.

[37] F. Guo, Y. Mu, W. Susilo, and V. Varadharajan, "Membership encryption and its applications," in 18th Australasian Conf. on Information Security and Privacy (ACISP2013), ser. LNCS, C. Boyd and L. Simpson, Eds., vol. 7959. Springer, 2013, pp. 219–234.

[38] S.-D. Kamvar, M.-T. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in 12th Conf. on World Wide Web (WWW'03). ACM, 2003, pp. 640–651.

[39] R. Zhou and K. Hwang, "Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing," Transactions on Parallel and Distributed Systems, vol. 18, no. 4, 2007, pp. 460–473.

[40] M. Gupta, P. Judge, and M. Ammar, "A reputation system for peer-to-peer networks," in 13th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'03). ACM, 2003, pp. 144–152.

Integrated Technologies for Communication Security and Secure Deletion on Android Smartphones

Alexandre Melo Braga, Daniela Castilho Schwab, Eduardo Moraes de Morais,
Romulo Zanco Neto, and André Luiz Vannucci
Centro de Pesquisa e Desenvolvimento em Telecomunicações (Fundação CPqD)
Campinas, São Paulo, Brazil
{ambraga,dschwab,emorais,romulozn,vannucci}@cpqd.com.br

***Abstract*— Nowadays, mobile devices are powerful enough to accomplish most of the tasks previously accomplished only by personal computers; that includes, for example, file management and instant messaging. On the other hand, in order to protect final user's interests, there is also an increasing need for security hardenings on ordinary, off-the-shelf devices. In fact, there is a need for practical security technologies that work at the application level, above the operating system and under the control of the user. This technology has to be easy to use in everyday activities and easily integrated into mobile devices with minimal maintenance and installation costs. The main contribution of this paper is to describe design and implementation issues concerning the construction of an integrated framework for securing both communication and storage of sensitive information of Android smartphones. Four aspects of the framework are detailed in this paper: the construction of a cryptographic library, its use in the development of a cryptographically secure instant message service, the integration with an encrypted file system, and the addition of secure deletion technologies. Also, an analysis of non-standard cryptography is provided, as well as performance evaluation of a novel secure deletion technique. The proposed framework is supposed to work in user-mode, as an ordinary group of mobile apps, without root access, with no need for operating system modification, in everyday devices.**

Keywords-cryptography; surveillance; security; Android; instant message; secure deletion; secure storage; encrypted file system; flash memory.

I. INTRODUCTION

Nowadays, the proliferation of smartphones and tablets and the advent of cloud computing are changing the way people handle their personal, maybe private, information. In fact, many users keep their sensitive data on mobile devices as well as on cloud servers.

The current generation of mobile devices is powerful enough to accomplish most of the tasks previously accomplished only by personal computers. That includes, for example, file management operations (such as create, read, update, and delete) and instant message capabilities. Also, today's devices possess operating systems that are hardware-agnostic by design and abstract from ordinary users all hardware details, such as writing procedures for flash memory cards.

However, there is no free lunch, and mobile devices, as any other on-line computer system, are vulnerable to many kinds of data leakage. Unfortunately, as the amount of digital data in mobile devices grows, so does the theft of sensitive data through loss of the device, exploitation of vulnerabilities or misplaced security controls. Sensitive data may also be leaked accidentally due to improper disposal of devices.

Contemporary to this paradigm shift from ordinary computers to mobile devices, the use in software systems of security functions based on cryptographic techniques seems to be increasing as well, maybe as a response to the new security landscape. The scale of cryptography-based security in use today seems to have increased not only in terms of volume of encrypted data, but also relating to the amount of applications with cryptographic services incorporated within their functionalities. In addition to the traditional use cases historically associated to stand-alone cryptography (e.g., encryption/decryption and signing/verification), there are new application-specific usages bringing diversity to the otherwise known threats to cryptographic software.

For example, today's secure phone communication does not mean only voice encryption, but encompasses a plethora of security services built over the ordinary smartphone capabilities. To name just a few of them, these are SMS encryption, Instant Message (IM) encryption, voice and video chat encryption, secure conferencing, secure file transfer, secure data storage, secure application containment, and remote security management on the device, including management of cryptographic keys. It is not surprisingly that, with the increasing use of encryption systems, an attacker wishing to gain access to sensitive data is directed to weaker targets. On mobile devices, one such attack is the recovery of supposedly erased data from internal storage, possibly a flash memory card. Also, embedded security technologies can suffer from backdoors or inaccurate implementations, in an attempt to facilitate unauthorized access to supposedly secure data.

This paper describes design and implementation issues concerning the construction of an integrated framework for securing both communication and storage of sensitive information of Android smartphones. Preliminary versions of this work have been addressed in previous publications [1][2][3], as part of a research project [4][5][6] targeting security technologies on off-the-shelf mobile devices.

Additionally, it is a real threat the misuse of security standards by intelligence agencies. The motivation behind

the special attention given to the selection of cryptographic algorithms lies in the recently revealed weakness, which may be intentionally included by foreign intelligence agencies, in international encryption standards [7][8]. This fact alone raises doubt on all standardized algorithms, which are internationally adopted. In this context, a need arose to treat what has been called "alternative" or "non-standard" cryptography in opposition to standardized cryptography.

This work contributes to the state of the practice by discussing the technical aspects and challenges of cryptographic implementations, as well as their integration into security-ware applications on modern, Android-based, mobile devices. The main contributions of this paper are the following:

- Discuss the construction of a cryptographic library for Android devices, which focuses on design decisions as well as on implementation issues of both standard and non-standard algorithms;
- Describe the construction of a mobile application for secure instant messaging that uses the cryptographic library and is integrated with an encrypted file system;
- Describe an encrypted file-system that uses the cryptographic library and integrates secure deletion technologies;
- Propose and analyze new approaches to secure deletion of stored data on off-the-shelf mobile devices.

The remaining parts of the text are organized as follows. Section II offers background on the subject. Section III presents related work. Section IV treats the construction of the secure chat. Section V details the constructions of the cryptographic library. Section VI describes the encrypted file system with secure deletion. Section VII discusses integration aspects. Section VIII concludes this text.

II. BACKGROUND

This section offers background information in the following selected subjects of interest: Android and Java technologies; cryptography issues in mobile devices; and secure storage and data deletion in flash memories.

A. General concepts for Android and Java

This section briefly describes the following topics: the Java Cryptographic Architecture (JCA) as a framework for pluggable cryptography; the Java Virtual Machine (JVM) along with its Garbage Collector (GC) and Just-in-Time (JiT) compilation; and The Dalvik Virtual Machine (DVM).

1) JCA

The JVM is the runtime software ultimately responsible for the execution of Java programs. In order to be interpreted by JVM, Java programs are translated to bytecodes, an intermediary representation that is neither source code nor executable. The JCA [9] is a software framework for use and development of cryptographic primitives in the Java platform. JCA defines, among other facilities, Application Program Interfaces (APIs) for digital signatures and secure hash functions [9]. On the other hand, APIs for encryption, key establishment and message authentication codes (MACs) are defined in the Java Cryptography Extension (JCE) [10].

The benefit of using a software framework, such as JCA, is to take advantage of good design decisions, reusing the whole architecture. The API keeps the same general behavior regardless of specific implementations. The addition of new algorithms is facilitated by the use of a standard API [11].

2) Garbage Colletion and JiT Compilation

An architectural feature of the JVM has great influence in the general performance of applications: the GC [12][13]. Applications have different requirements for GC. For some applications, pauses during garbage collection may be tolerable, or simply obscured by network latencies, in such a way that throughput is an important metric of performance. However, in others, even short pauses may negatively affect the user experience.

One of the most advertised advantages of JVM is that it shields the developer from the complexity of memory allocation and garbage collection. However, once garbage collection is a major bottleneck, it is worth understanding some aspects of its implementation.

Another important consideration on performance of Java programs is the JiT Compilation [12][14]. Historically, Java bytecode used to be fully interpreted by the JVM and presented serious performance issues. Nowadays, JiTC not only compiles Java programs, but also optimizes them, while they execute. The result of JiTC is an application that has portions of its bytecode compiled and optimized for the targeted hardware, while other portions are still interpreted. It is interesting to notice that JVM has to execute the code before to learn how to optimize it.

Unfortunately, there is a potential negative side to security in the massive use of JiT Compilation. Security controls put in place into source code, in order to avoid side-channels, can be cut off by JiT optimizations. JiTC is not able to capture programmer's intent that is not explicitly expressed by Java's constructs. That is exactly the case of constant time computations needed to avoid timing attacks. Security-ware optimizations should be able to preserve security decisions and not undo protections, when transforming source code for cryptographic implementations to machine code. Hence, to achieve higher security against this kind of attacks, it is not recommended to use JiTC technology, what constitutes a trade-off between security and performance. Further discussion of cryptographic side-channels and its detection in Java can be found in [15].

3) DVM

The DVM [16] is the virtual hardware that executes Java bytecode in Android. DVM is quite different from the traditional JVM, so that software developers have to be aware of those differences, and performance measurements over a platform independent implementation have to be taken in both environments.

Compared to JVM, DVM is a relatively young implementation and did not suffered extensive evaluation. In fact, the first independent evaluation of DVM was just recently published [17]. There are three major differences between DVM and JVM. First of all, DVM is a register-based machine, while JVM is stack-based. Second, DVM applies trace-based JiTC, while JVM uses method-based

JiTC. Finally, former DVM implementations use mark-and-sweep GC, while current JVM uses generation GC.

Also, results from that DVM evaluation [17] suggest that current implementations of DVM are slower than current implementations of JVM. Concerning cryptographic requirements, a remarkable difference between these two environments is that the source of entropy in DVM is significantly different from the one found on JVM.

B. Security and cryptography issues on Android devices

A broad study on Android application security, especially focused on program decompilation and source code analysis, was performed by [18]. There are several misuse commonly found on cryptographic software in use today. According to a recent study [19], the most common misuse of cryptography in mobile devices is the use of deterministic encryption, where a symmetric cipher in Electronic Code Book (ECB) mode appears mainly in two circumstances: Advanced Encryption Standard (AES) in ECB mode of operation (AES/ECB for short) and Triple Data Encryption Standard in ECB mode (TDES/ECB). A possibly worse variation of this misuse is the Rivest-Shamir-Adleman (RSA) cryptosystem in Cipher-Block Chaining (CBC) mode with Public-Key Cryptography Standards Five (PKCS#5) padding (without randomization) [20]. Another frequent misuse is hardcoded Initialization Vectors (IVs), even with fixed or constant values [20]. A related misuse is the bad habit of hardcoded seeds for PRNGs [19].

A common misunderstanding concerning the correct use of IVs arises when (for whatever reason) programmers need to change operation modes of block ciphers. For instance, the Java Cryptographic API [9] allows operation modes to be easily changed, but without considering IV requirements.

According to a NIST standard [21], CBC and Cipher feedback (CFB) modes require unpredictable IVs. However, Output feedback (OFB) mode does not need unpredictable IVs, but it must be unique to each execution of the encryption operation. Considering these restrictions, IVs must be both unique and unpredictable, in order to work interchangeably with almost all common operation modes of block ciphers. The Counter (CTR) mode requires unique IVs and this constraint is inherited by authenticated encryption with Galois/Counter mode (GCM) [22].

C. Secure storage and deletion on flash memory

Traditionally, the importance of secure deletion is well understood by almost everyone and several real-world examples can be given on the subject: sensitive mail is shredded; published government information is selectively redacted; access to top secret documents ensures all copies can be destroyed; and blackboards at meeting rooms are erased after sensitive appointments.

In mobile devices, that metaphor is not easily implemented. All modern file systems allow users to "delete" their files. However, on many devices the remove-file command misleads the user into thinking that her file has been permanently removed, when that is not the case. File deletion is usually implemented by unlinking files, which only changes file system metadata to indicate that the file is "deleted"; while the file's full content remains available in physical medium. This process is known as logical deletion.

Unfortunately, despite the fact that deleted data are not actually destroyed in the device, logical deletion has the additional drawback that ordinary users are generally unable to completely remove her files. On the other hand, advanced users or adversaries can easily recover logically deleted files.

Deleting a file from a storage medium serves two purposes: (i) it reclaims storage to operating system and (ii) ensures that any sensitive information contained in the file becomes inaccessible. The second purpose requires that files are securely deleted.

Secure data deletion can be defined as the task of deleting data from a physical medium so that the data is irrecoverable. That means its content does not persist on the storage medium after the secure deletion operation.

Secure deletion enables users to protect the confidentiality of their data if their device is logically compromised (e.g., hacked) or stolen. Until recently, the only user-level deletion solution available for mobile devices was the factory reset, which deletes all user data on the device by returning it to its initial state. However, the assurance or security of such a deletion cannot be taken for granted, as it is highly dependent on device's manufacturer. Also, it is inappropriate for users who wish to selectively delete data, such as some files, but still retain their address books, emails and installed applications.

Older technologies [23] claim to securely delete files by overwriting them with random data. However, due the nature of log-structured file systems used by most flash cards, this solution is no more effective than logically deleting the file, since the new copy invalidates the old one but does not physically overwrite it. Old secure deletion approaches that work at the granularity of a file are inadequate for mobile devices with flash memory cards.

Today, secure deletion is not only useful before discarding a device. On modern mobile devices, sensitive data can be compromised at unexpected times by adversaries capable of obtaining unauthorized access to it. Therefore, sensitive data should be securely deleted in a timely fashion.

Secure deletion approaches that target sensitive files, in the few cases where it is appropriate, must also address usability concerns. A user should be able to reliably mark their data as sensitive and subject to secure deletion. That is exactly the case when a file is securely removed from an encrypted file system. On the other hand, approaches that securely delete all logically deleted data, while less efficient, suffer no false negatives. That is the case for purging.

III. Related Work

This section discusses related work on following subjects: cryptography implementation on mobile devices, security of IM applications, and secure storage and deletion.

A. Cryptography implementation on mobile devices

A couple of years ago, the U.S. National Security Agency (NSA) started to encourage the use of off-the-shelf mobile devices, in particular smartphones with Android, for

communication of classified information [24]. The document fosters the adoption of two layers of cryptography for communication security. One is provided by infrastructure (e.g., VPN) and other implemented at the application layer.

Regarding the performance evaluation of cryptographic libraries on Android smartphones, there are tests made on the Android platform for the BouncyCastle and Harmony cryptographic libraries, both already available on the platform [25].

A few works could be found concerning efficient implementation of cryptography on smartphones. The first one [26] presented an efficient Java implementation of elliptic curve cryptography for J2ME-enabled mobile devices. That Java implementation has an optimized scalar multiplication that combines efficient finite-field arithmetic with efficient group arithmetic. A second work [27] presented an identity-based key agreement protocol for securing mobile telephony in GSM and UMTS networks. The paper proposes an approach to speed up client-side cryptography using server-aided cryptography, by outsourcing computationally expensive cryptographic operations to a high-performance backend computing server.

Another work [28] presents a Java port (jPBC) of the PBC library written in C, which provides simplified use of bilinear maps and supports different types of elliptic curves.

A recent study [6] showed that despite the observed diversity of cryptographic libraries in academic literature, this does not mean those implementations are publicly available or ready for integration with third party software. In spite of many claims on generality, almost all of them were constructed with a narrow scope in mind and prioritizes academic interest for non-standard cryptography. Furthermore, portability to Android used to be a commonly neglected concern on cryptographic libraries [6].

Recently, the European Union Agency for Network and Information Security (ENISA) has published two technical reports [29][30] about the correct and safe use of cryptography to protect private data in on-line system, giving attention to cloud and mobile environments. One report [29] focuses on algorithms, key size and parameters. Other report [30] gives attention to cryptographic protocols, and tries to point legacy issues and design vulnerabilities.

B. Security issues in IM protocols and applications

The work of Xuefu and Ming [31] shows the use of eXtensible Messaging and Presence Protocol (XMPP) for IM on web and smartphones. Massandy and Munir [32] have done experiments on security aspects of communication, but there are unsolved issues, such as strong authentication, secure storage, and implementation of good cryptography, as shown by Schrittwieser et al. [33].

It seems that the most popular protocol for secure IM in use today is the Off-the-Record (OTR) Messaging [34], as it is used by several secure IM apps. OTR Messaging handshake is based upon the SIGMA key exchange protocol [35], a variant of Authenticated Diffie-Hellman (ADH) [36], just like Station-to-Station (STS) [37][38].

A good example of security issues found in current IM software is a recently discovered vulnerability in WhatsApp [39]. The vulnerability resulting from misuse of the Rivest Cipher 4 (RC4) stream cipher in a secure communication protocol allowed the decryption, by a malicious third party able to observe conversations, of encrypted messages exchanged between two WhatsApp users.

In order to be fair, it is worth note that WhatsApp has recently announced an effort for hardening its communication security with end-to-end encryption [40].

C. Secure storage and deletion

This section briefly describes related work on the subjects of secure deletion and encrypted file systems on mobile devices, particularly Android.

The use of cryptography as a mechanism to securely delete files was first discussed by Boneh and Lipton [41]. Their paper presented a system which enables a user to remove a file from both file system and backup tapes on which the file is stored, just by forgetting the key used to encrypt the file.

Gutman [23] covered methods available to recover erased data and presented actual solutions to make the recovery from magnetic media significantly more difficult by an adversary. Flash memory barely existed at the time it was written, so it was not considered by him.

K. Sun et al. [42] proposed an efficient secure deletion scheme for flash memory storage. This solution resides inside the operating system and close to the memory card controller.

Diesburg and Wang [43] presented a survey summarizing and comparing existing methods of providing confidential storage and deletion of data in personal computing environments, including flash memory issues.

Wang et al. [44] present a FUSE (File-system in USErspace) encryption file system to protect both removable and persistent storage on devices running the Android platform. They concluded that the encryption engine was easily portable to any Android device and the overhead due to encryption is an acceptable trade-off for achieving the confidentiality requirement.

Reardon et al. [45]-[49] have shown plenty of results concerning both encrypted file system and secure deletion. First, Reardon et al. [45] proposed the Data Node Encrypted File System (DNEFS), which uses on-the-fly encryption and decryption of file system data nodes to efficiently and securely delete data on flash memory systems. DNEFS is a modification of existing flash file systems or controllers that extended a Linux implementation and was integrated in Android operating system, running on a Google Nexus One smartphone.

Reardon et al. [46] also propose user-level solutions for secure deletion in log-structured file systems: purging, which provides guaranteed time-bounded deletion of all data previously marked to be deleted, and ballooning, which continuously reduces the expected time that any piece of deleted data remains on the medium. The solutions empower users to ensure the secure deletion of their data without

relying on the manufacturer to provide this functionality. These solutions were implemented on an Android smartphone (Nexus One).

In two recent papers, Reardon et al. [47][48] study the issue of secure deletion in details. First, in [47], they identify ways to classify different approaches to securely deleting data. They also describe adversaries that differ in their capabilities, show how secure deletion approaches can be integrated into systems at different interface layers. Second, in [48], they survey the related work in detail and organize existing approaches in terms of their interfaces to physical media. More recently, Reardon et al. [49] presented a general approach to the design and analysis of secure deletion for persistent storage that relies on encryption and key wrapping.

Finally, Skillen and Mannan [50] designed and implemented a system called Mobiflage that enables plausibly deniable encryption (PDE) on mobile devices by hiding encrypted volumes within random data on a device's external storage. They also provide [51] two different implementations for the Android OS to assess the feasibility and performance of Mobiflage: One for removable SD cards and other for internal partition for both apps and user accessible data.

The above mentioned works suffer from at last one of the following disadvantages:

- Requires modification of the host operating system or device, so the solution does not work on off-the-shelf devices without modification of OS internals;
- Limits the available (free) storage to ordinary applications, possibly leading apps to starvation by lack of storage;
- Inserts abnormal behavior to storage usage that can potentially slow down the whole system, when using incremental memory sweeping by a single-file, single-thread application.

The secure deletion approach proposed in this paper provides alternative solutions to these disadvantages.

IV. CONSTRUCTION OF A SECURE CHAT APPLICATION

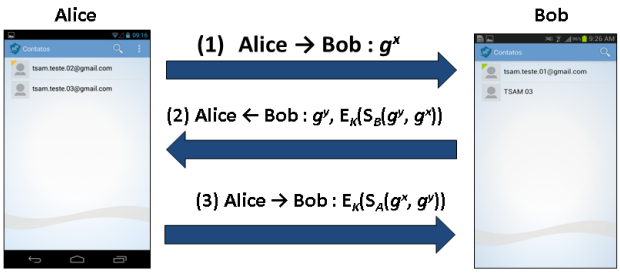
This section describes design and implementation issues concerning the construction of CryptoIM, a prototype app for cryptographically secure, end-to-end communication, which operates on a device-to-device basis, exchanging encrypted instant messages via standard transport protocols.

A. Cryptographic services for IM applications

CryptoIM implements the basic architecture used by all IM applications, using the standard protocol XMPP [52] at the transport layer. The application then adds a security layer to XMPP, which is composed of a protocol for session key agreement and cryptographic transaction to transport encrypted messages. The security negotiation is indeed a protocol for key agreement, as illustrated by Figure 1.

To accomplish cryptographically secure communication, Alice and Bob agree on the following general requirements:

- An authentication mechanism of individual messages;
- An encryption algorithm and modes of operation;
- A key agreement protocol;



E_K → Encryption with key k. S_B or S_A → Sign with Bob's or Alice's private key.

Figure 1. Station to Station (STS) protocol.

- A mechanism to protect cryptographic keys at rest.

To avoid known security issues in instant messaging applications [33][39], the key agreement protocol must provide the following security properties [53]:

- Mutual authentication of entities;
- Mutually authenticated key agreement;
- Mutual confirmation of secret possession;
- Perfect Forward Secrecy (PFS).

As a general goal, the CryptoIM is intended to be used in the protection of cryptographically secure communication via mobile devices. In order to be useful, the underlying cryptographic library had to accomplish a minimum set of functional requirements.

Once JCA [9] was defined as the architectural framework, as it is the standard API for cryptographic services on Android, the next design decision was to choose the algorithms minimally necessary to implement a scenario of secure communication via mobile devices. The choice of a minimalist set was an important design decision in order to provide a fully functional Cryptographic Service Provider (CSP) in a relatively short period of time. This minimalist construction had to provide the following set of functions:

a) A symmetric algorithm to be used as block cipher, along with the corresponding key generation function, and modes of operation and padding;
b) An asymmetric algorithm for digital signatures, along with the key-pair generation function. This requirement brings with it the need for some sort of digital certification of public keys;
c) A one-way secure hash function. This is a support function to be used in MACs and signatures;
d) A Message Authentication Code (MAC), based on a secure hash or on a block cipher;
e) A key agreement mechanism or protocol to be used by communicating parties that have never met before, but need to share an authentic secret key;
f) A simple way to keep keys safe at rest and that does not depend on hardware features;
g) A Pseudo-Random Number Generator (PRNG) to be used by key generators and nonce generators.

The cryptographic library supporting CryptoIM was designed to meet each one of these general requirements, resulting in an extensive implementation.

B. *Advanced cryptographic features*

Three improvements to CryptoIM were necessary to integrate it to other apps in the framework. The first is a mobile PKI for digital certification, which is fully integrated to the mobile security framework. PKI's Server-side is based upon the EJBCA [54]. Client-side follows recommendations for handling certificates on mobile devices [55].

The second is a secure text conference (or group chat) via instant messages. As depicted in Figure 2, the Organizer or Chair of the conference requests the conference creation to the Server, as this is an ordinary XMPP feature. The key agreement for the requested conference proceeds as follows, where $Enc_k(x)$ means encryption of x with key k:

1. Chair (C) creates the key for that conference (c_k);
2. For each guest (g[i]), Chair (C) does:
   a) Opens a STS channel with key k: C ↔ g[i], key k;
   b) Sends c_k on time t to g[i]: C → g[i]: $Enc_k(c_k)$.

These steps constitute a point-to-point key transport using symmetric encryption, which is carried out by STS protocol. After that, all guests share the same group key and conference proceeds as a multicast of encrypted messages.

The third improvement is a secure file transfer that is fully integrated to the encrypted file system described in Section VI. Figure 3 illustrated the secure transfer as a step-by-step procedure. The encrypted file system and its file management tool are jointly referred as CryptoFM. The eleven steps for secure file transfer are as follows:

1. Alice activates the file transfer function;
2. Alice's CryptoIM activates the local instance of CryptoFM and passes to it the key K_{FT} (key derived from K_{STS} conversation) for secure transport of files;

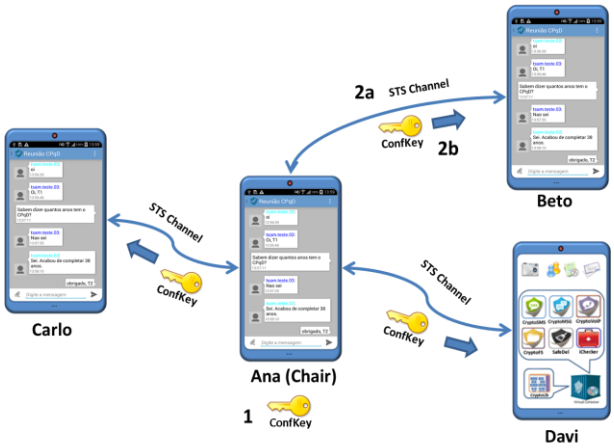


Figure 2. Key agreement for secure conference.

3. Alice chooses, from her CryptoFM, the file to be transferred and exports it from encrypted file system;
4. The exported file is encrypted with the key K_{FT} and stored in a public folder;
5. CryptoIM gets the encrypted file from public folder;
6. The encrypted file and related metadata are transmitted from Alice to Bob through a secure channel (STS channel) over XMPP;
7. The file is received by Bob, who accepts the transfer in his CryptoIM and saves the file;
8. The encrypted file is temporarily saved in a public folder recognized by the Bob's CryptoFM;
9. Bob's CryptoIM activates its local CryptoFM and passes to it the key K_{FT} (key derived from K_{STS}

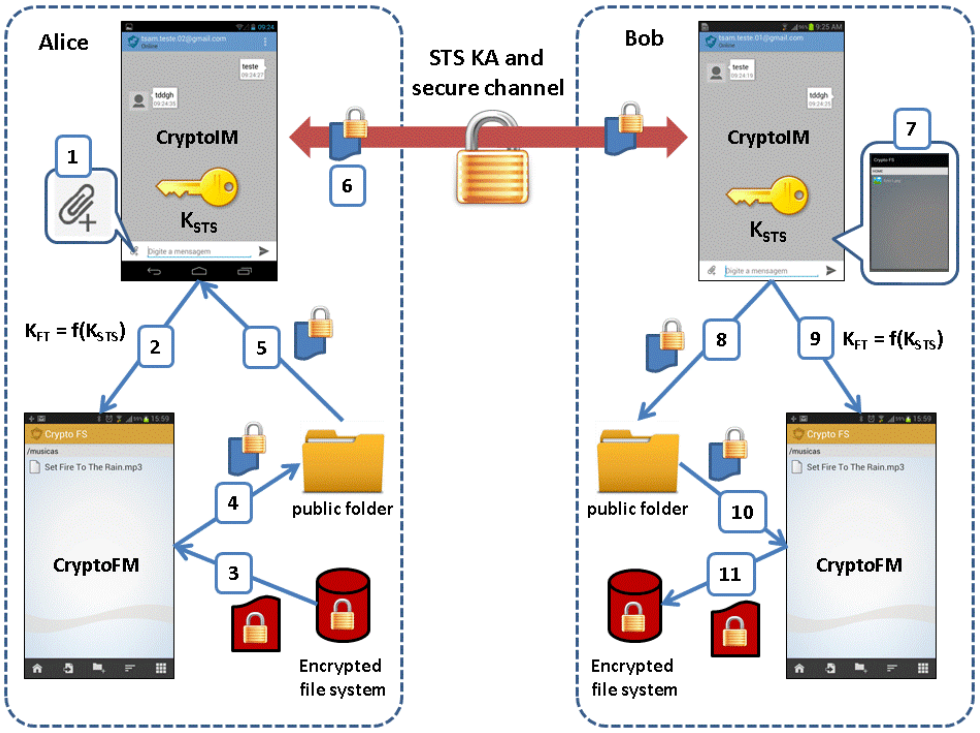


Figure 3. Secure file transfer is integrated to both CryptoIM and CryptoFM.

conversation) used to securely transport of the file;

10. Bob's CryptoFM, in a secure import operation, gets the encrypted file from the public folder and decrypts with key K_{FT};
11. CryptoFM saves the received file into its encrypted file system.

This procedure has the possible vulnerability of leaving temporary files or residual, unencrypted information at local storage. This vulnerability can show up at both sides of file transfer. In fact, this issue raised the need for a method for secure deletion and memory purging.

In summary, the three remarkable differences between CryptoIM and the related work are the following. First, the prototype uses STS protocol and its variants to accomplish authenticated key agreement. This has the benefit of facilitating protocol extension to use alternative cryptographic primitives. Also, STS is used as building block for both multi-user conference and secure file transfer. Second, authenticated encryption is the preferred encryption mechanism to protect messages, so the burden of IV management is minimized. Third, it is fully integrated to an encrypted file system.

V. Construction of a Cryptographic Library

This section describes both the design decisions and implementation issues concerning the construction of a cryptographic library for Android devices. This library support all secure apps included in the secure framework, including CryptoIM, a secure chat detailed in Section IV, and CryptoFM, an encrypted file-system introduced in Section IV and detailed in Section VI.

Four aspects of the implementation were discussed in this paper: selection of cryptographic primitives, architecture of components, performance evaluation on Android devices, and the implementation of non-standard cryptographic algorithms.

As previously stated, a need arose to treat what has been called "alternative" or "non-standard" cryptography in opposition to standardized cryptographic schemes. The final intent was strengthening the implementation of advanced cryptography and fostering their use. Non-standard cryptography provides advanced mathematical concepts, such as bilinear pairings and elliptic curves, which are not fully standardized by foreign organizations, and suffer constant improvements.

In order to facilitate the portability of the cryptographic library for mobile devices, in particular for the Android platform, the implementation was performed according to standard cryptographic API for Java, the JCA [9][56], its name conventions [57], and design principles [10][58].

Once JCA was defined as the architectural framework, the next design decision was to choose the algorithms minimally necessary to a workable cryptographic library. The current version of this implementation is illustrated by Figure 4 and presents the cryptographic algorithms and protocols described in the following paragraphs. The figure shows that frameworks, components, services and applications are all on top of JCA API. The Cryptographic Service Provider (CSP) is in the middle, along with

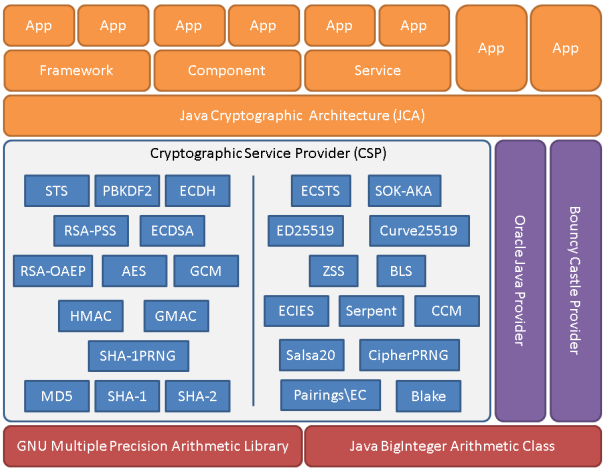


Figure 4. Cryptographic Service Provider Architecture.

BouncyCastle and Oracle providers. Arithmetic libraries are at the bottom.

Figure 4 shows the CSP divided in two distinct cryptographic libraries. The left side shows only standardized algorithms and comprises a conventional cryptographic library. The right side features only non-standard cryptography and is an alternative library. The following subsections describe these two libraries.

A. Standard cryptography

This subsection details the implementation choices for the standard cryptographic library. The motivations behind this implementation were all characteristics of standardized algorithms: interoperability, documentation, and testability. The standard cryptography is packaged as a pure-Java library according to the JCA specifications.

The block cipher is the AES algorithm, which was implemented along with the modes of operation: ECB, and CBC [21], as well as the GCM mode for authenticated encryption [22]. PKCS#5 [59] is the simplest padding mechanism and was chosen for compatibility with other CSPs. As GCM mode for authenticated encryption only uses AES encryption, the optimization of encryption received more attention than AES decryption. Implementation aspects of AES and other cryptographic algorithms can be found on literature [60][61][62], in particular [63].

The Signature algorithm is the RSA-PSS that is a Probabilistic Signature Scheme (PSS) constructed over the RSA signature algorithm. RSA-PSS is supposed to be more secure than ordinary RSA [62][64]. Asymmetric encryption is provided by the RSA-OAEP [62][64].

Two cryptographically secure hashes were implemented, SHA-1 [65] and MD5. It is well known by now that MD5 is considered broken and is not to be used in serious applications, it is present for ease of implementation. In current version, there is no intended use for these two hashes. Their primary use will be as the underling hash function in MACs, digital signatures and PRNGs. The Message Authentication Codes chosen were the HMAC [66] with SHA-1 and SHA2 as the underling hash functions, and the

GMAC [22], which can be directly derived from GCM mode. SHA-2 family of secure hashes supplies the need for direct use of single hashes.

The need for key agreement was fulfilled by the Station-to-Station (STS) protocol, which is based upon Authenticated Diffie-Hellman [36], and provides mutual key authentication and confirmation [37][38].

Finally, the mechanism for Password-based Encryption (PBE) is based on the Password-Based Key Derivation Function 2 (PBKDF2) [59], and provides a simple and secure way to store keys in encrypted form. In PBE, a key-encryption-key is derived from a password.

B. Non-standard cryptography

This subsection details the implementation choices for the alternative cryptographic library. The non-standard cryptography is a dynamic library written in C and accessible to Java programs through a Java Native Interface (JNI) connector, which acts as a bridge to a JCA adapter.

Some of the constructs are based upon a reference implementation [67]. The most advanced cryptographic protocols currently implemented are listed below:

a) Curve25519 [68] is used to provide a key agreement protocol equivalent to the Elliptic Curve Diffie–Hellman (ECDH) [69], but over a non-standard curve. The key agreement protocol ECDH is a variation of the Diffie-Hellman (DH) protocol using elliptic curves as the underlying algebraic structure;
b) ED25519 [70] is utilized to construct a digital signature scheme that corresponds to the Elliptic Curve Digital Signature Algorithm (ECDSA) [71], but over a non-standard curve that is birationally equivalent to Curve25519. ECSS [69] is a variation of ECDSA that does not require the computation of inverses in the underlying finite field, obtaining a signature algorithm with better performance;
c) Sakai-Ohgishi-Kasahara (SOK) [72]. This protocol is a key agreement for Identity-Based Encryption (IBE). Sometimes, it is called SOKAKA for SOK Authenticated Key Agreement;
d) Boneh-Lynn-Shacham (BLS) [73]. A short digital signature scheme in which given a message m, it is computed S = H (m), where S is a point on an elliptic curve and H() is a secure hash;
e) Zhang-Safavi-Susilo (ZSS) [74]. Similar to the previous case, it is a more efficient short signature, because it utilizes fixed-point multiplication on an elliptic curve rather arbitrary point;
f) Blake [75]. Cryptographic hash function submitted to the worldwide contest for selecting the new SHA-3 standard and was ranked among the five finalists;
g) Elliptic Curve Integrated Encryption Scheme (ECIES) [69]. This is an asymmetric encryption algorithm over elliptic curves. This algorithm is non-deterministic and can be used as a substitute of the RSA-OAEP, with the benefit of shorter cryptographic keys;
h) Elliptic Curve Station-to-Station (ECSTS) [69]. Variation of STS protocol using elliptic curves and ECDH as a replacement for ADH;
i) Salsa20 [76]. This is a family of 256-bit stream ciphers submitted to the ECRYPT Project (eSTREAM);
j) Serpent [77]. A 128-bit block cipher designed to be a candidate to contest that chose the AES. Serpent did not win, but it was the second finalist and enjoys good reputation in the cryptographic community;
k) CipherPRNG based upon the construction described by Petit et al. [78], which offers protection against side channel attacks. There is a security proof that the scheme produces a sequence of random numbers indistinguishable from the uniform distribution.

C. Security decisions for non-standard cryptography

Among the characteristics that were considered in the choice of alternative cryptographic primitives, side channels protection was a prevailing factor and had distinguished role in the design of the library. For instance, schemes with known issues were avoided, while primitives that were constructed to resist against such attacks are currently being regarded for inclusion in the architecture. Furthermore, constant-time programming techniques, like for example in table accessing operations for AES, are being surveyed in order to became part of the implementation.

Concerning mathematical security of non-standard cryptography, the implementation offers alternatives for 256-bit security for both symmetric and asymmetric encryption. For instance, Serpent-256 corresponds to AES-256 block cipher, while the same security level is achieved in asymmetric world using elliptic curves over 521-bit finite fields, what can only be possible in standard cryptography using 15360-bit RSA key size. Thus, in higher security levels, non-standard primitives performance is significantly improved in relation to standard algorithms, but an extensive analysis of this scenario, with concrete timing comparisons, is left as future work.

Short signatures, such as BLS and ZSS (BBS), are not as fast as EC, since this kind of constructions are based on bilinear pairings. Here, there is a tradeoff, because the signature can be roughly half the size of a regular ECDSA signature, but the verification algorithm must compute a bilinear pairing and, therefore, is less efficient. It is important to remark that the ability to compute bilinear pairings allows us to achieve many new cryptographic functionalities, such as identity based cryptography and certificateless encryption. Furthermore, the scheme ED25519 is a recently proposed digital signature cryptosystem that has been built over elligator curves [70], which offers advantages against side channel attacks and is a non-standard construction which may not be susceptible to surveillance manipulation.

A final remark about the use of non-standard cryptography is that working with advanced cryptographic techniques that have not been sufficiently analyzed by the scientific community has its own challenges and risks. There are occasions when the design of a non-standard

cryptographic library has to be conservative in order to preserve security.

For instance, a recent improvement in mathematics [79][80] had eliminated an entire line of research in theoretical cryptography. Such advancement affected elliptic curve cryptography using a special kind of binary curves called supersingular curves, but had no effect on the bilinear pairings over primes fields or encryption on ordinary (common) binary curves. Thus, these two technologies remain cryptographically secure. Unfortunately, the compromised curves were in use and had to be eliminated from the cryptographic library.

As pairings on prime fields can still be securely used in cryptographic applications, the implementation was adapted to that new restricted context. Additionally, ordinary elliptic curves may still be used for cryptographic purposes, considering they are not supersingular curves, and the implementation had to adapt to that fact, too.

D. Performance Evaluation

Performance evaluation of Java programs, either in standard JVM or DVM/Android, is a stimulating task due to many sources of interference that can affect measurements. As discussed in previous sections, GC and JiTC have great influence over the performance of Java programs. For instance, Garbage Collections (GC) as well as optimizations and recompilations can be clearly identified in diagrams, as shown in Figure 5(A). The figure shows the time consumed by the first 300 executions of a pure-Java implementation of the AES algorithm, for both encryptions (E) and decryptions (D) of a small block of data, with a 128-bit key. The measurements were taken on a Samsung Galaxy S III (Quad-core 1.4 GHz Cortex-A9 processor, 1GB of RAM, and Android 4.1). The figure shows that at the very first moments of execution, the algorithm has a relatively poor performance, since the bytecode is been interpreted, analyzed for optimizations, and compiled at the same time. After this short period, the overall performance of the application improves and the execution tends to stabilize at an acceptable level of performance, despite a few GC calls.

Due to the above mentioned limitations, two approaches of measurement have been used for the evaluation of cryptographic functions. The first one was the measurement of elapsed time for single cryptographic functions processing a single (small) block of data. This approach suffers from the interference of GC and JiTC. The JiTC interference can be eliminated by discarding all the measurements collected before code optimization. The GC interference cannot be completely eliminated, though.

Figure 5(B) exemplifies the first approach and shows the comparative performance of AES's encryptions (E) and decryptions (D) of a single block of data, for two cryptographic providers for Android: this CryptoLib (CSP), and BouncyCastle's [81] deployment for Android, SpongeCastle (SC) [82]. AES were setup to ECB mode and 128-bit key. The measurements were taken on a smartphone Samsung Galaxy S III (Quad-core 1.4 GHz Cortex-A9 processor, 1GB of RAM, and Android 4.1). The procedure consisted of processing a single block of data in a loop of 10,000 iterations.

In order to inhibit the negative influence of GC and JiTC, two metrics were taken: the average of all iterations and the 9th centile. None of them resulted in a perfect metric, but the 9th centile were able to reduce negative influence from GC and JiTC. For small data chunks, CSP is faster than SC.

The second approach for performance evaluation supposes that final users of mobile devices will not tuning their Java VMs with obscure configuration options in order

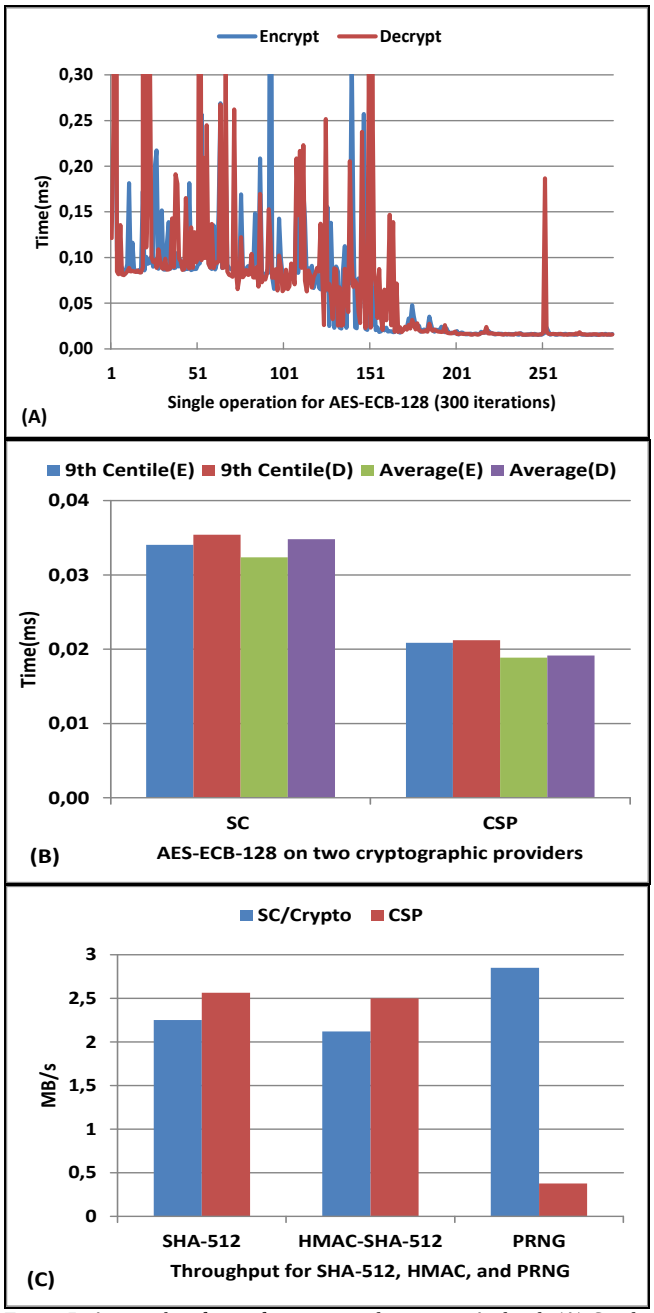


Figure 5. Approaches for performance evaluation on Android. (A) Single measurements suffer from GC and JiTC. (B) 9th centile and average show expected behavior, but are less accurate. (C) Throughput gives user's perceived responsiveness.

to achieve maximum performance. On the contrary, almost certainly, they will use default configurations, with minor changes on device's settings. Thus, the responsiveness of an application tends to be more relevant to final users than the performance of single operations.

The second approach of measurement takes into account the responsiveness of cryptographic services and considers the velocity with which a huge amount of data can be processed, despite the interferences of GC and JiTC. The amount of work performed per unit of time is called the throughput of the cryptographic implementation.

Figure 5(C) shows the throughput of SHA-256 and HMAC-SHA-256 implemented by CryptoLib (CSP) compared to SC. Also it shows the throughput for two instances of Pseudo Random Number Generator (PRNG): CSP's CipherPRNG compared to a SHA1PRNG available to Android apps through a provider called Crypto. The measurements were taken on a smartphone of type Samsung Galaxy S III (Quad-core 1.4 GHz Cortex-A9 processor, 1GB of RAM, and Android 4.1).

The procedure consisted of processing an input file of 5 MB, in a loop of 500 iterations. It is interesting to observe that CSP and SC are quite similar in performance for SHA-256 and HMAC, CSP is slightly better. However, CSP's CipherPRNG has shown a low throughput, mostly because its construction is relatively inefficient, since it is based on block ciphers instead of hash functions. Nonetheless, this implementation is still a proof of concept and better timings are expected in the future.

Performance measurements for other implementations of non-standard cryptography were taken as well. Despite been implemented in C and not been subjected to GC and JiTC influences, non-standard cryptography usually has no standard specifications or safe reference implementations. Neither it is in broad use by other cryptographic libraries. Because of that, comparisons among implementations of the same algorithm are barely possible. On the other hand, it is feasible to compare alternative and standard cryptography, considering the same type of service.

For the non-standard cryptography implementations, performance measurements were taken in two smartphones: (i) LG Nexus 5 with processor 2.3 GHz quad-core Krait 400, 2GB of RAM, and 16GB of storage and (ii) Samsung Galaxy S III with processor of 1.4 GHz quad-core Cortex-A9, 1 GB

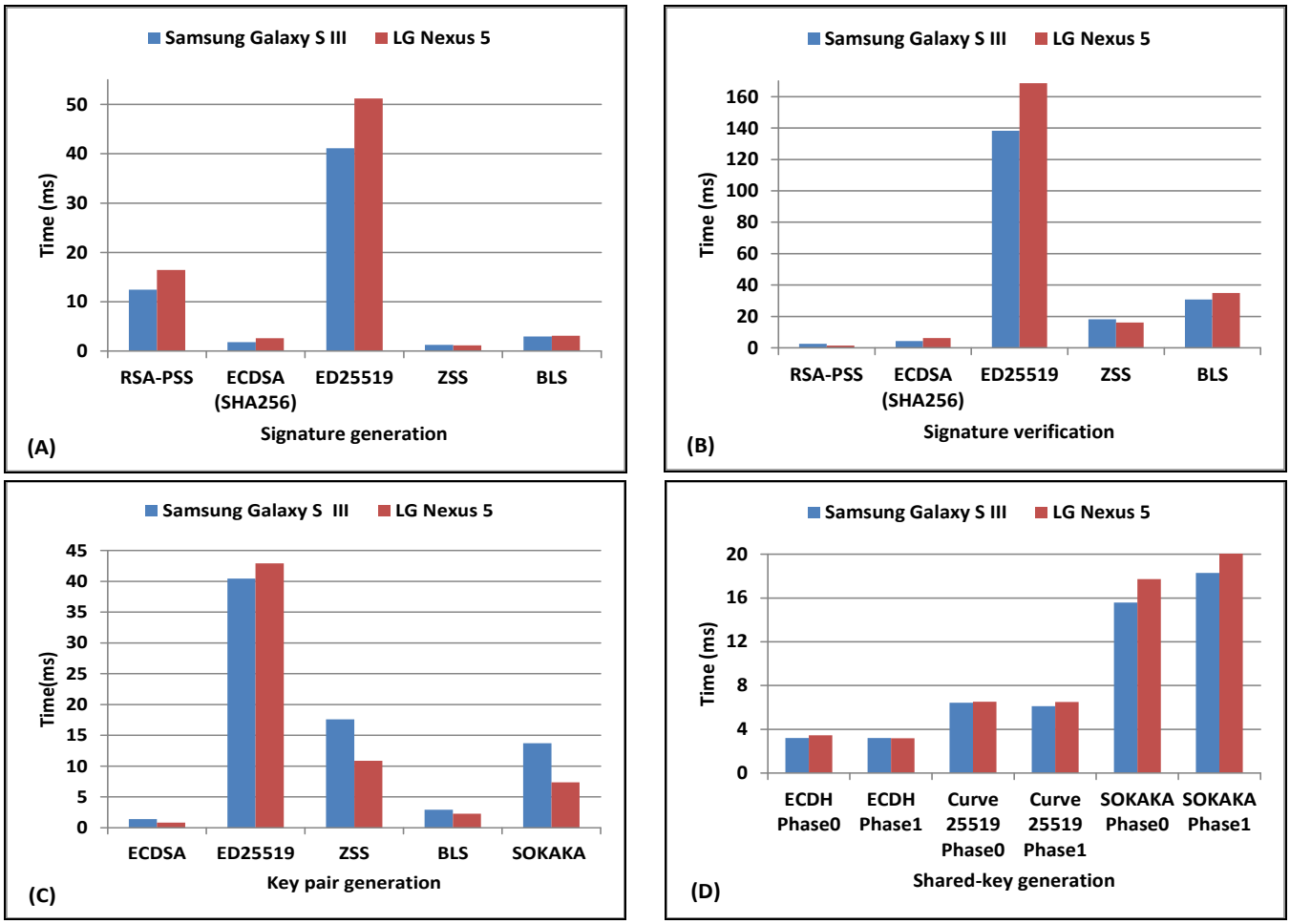


Figure 6. Performance evaluation of non-standard cryptography compared to standards. RSA uses 1024-bit key, all others have security level of 256-bit. Digital signatures: (A) generation, (B) verification, and (C) key pair generation. Key Agreement: (D) parameters generation and secret agreement.

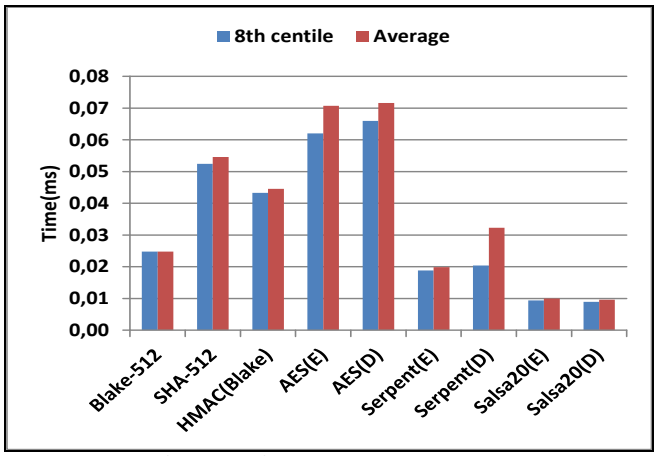


Figure 7. Performance of non-standard cryptography (symmetric encryption, secure hash, and MACs) compared to AES and SHA-512.

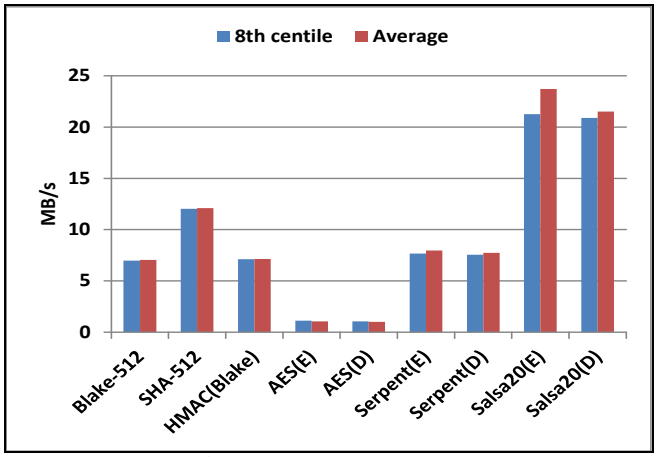


Figure 8. Throughput of non-standard cryptography (symmetric encryption, secure hash, and MACs) compared to AES and SHA-512.

of RAM, and 16 GB of storage.

Figure 6 shows two types of services: digital signatures at the top and key agreement (KA) at the bottom. The bar chart Figure 6(A) shows generation of digital signatures for five algorithms: RSA (1024-bit key), ECDSA (with SHA-256), ED25519, BLS and ZSS (BBS), all of them for 256-bit security. Traditionally, RSA is the slowest one. Elliptic curve cryptography, as in ECDSA, is faster. Short signatures, such as BLS and ZSS (BBS), are not as fast as EC. The scheme ED25519 is the slowest one. This implementation is still a proof of concept and better timings are expected after optimizations.

Bar chart of Figure 6(B) shows verification of digital signatures for five algorithms: RSA (1024-bit key), ECDSA (with SHA-256), ED25519, BLS and ZSS (BBS), with 256-bit security. Traditionally, RSA verification is the fastest one. Elliptic curve cryptography, as in ECDSA, is not that fast. Short signatures, such as BLS and ZSS (BBS), are terribly slow, due to complex arithmetic involved in bilinear pairings computations. ED25519 is the slowest one.

Figure 6(C) shows key pair generation for ED25519, BLS, ZSS (BBS) and SOKAKA, a pairings-based KA scheme, compared to ECDSA. Again, performance is slow for BLS, ZSS (BBS), and SOKAKA. ED25519 is the slowest. Figure 6(D) shows two KA schemes (Curve25519 and SOKAKA) compared to ECDH. ECDH is quite fast. Curve25519 is faster than SOKAKA.

Additional measurements were taken for symmetric, non-standard algorithms on the same Samsung Galaxy S III. Figure 7 shows time measurements of single-block operations for the following algorithms: (i) Blake 512 and HMAC with Blake compared to SHA-512; (ii) Serpent and Salsa20 compared to AES. Algorithms were setup with a 256-bit key, if needed. The bar chart shows both average and the 8th centile of 10 thousand operations. Serpent is faster than homegrown AES, but Salsa20 is the fastest. Blake 512 is quite competitive to SHA-512 for small amounts of data.

Figure 8 tries to capture the perceived responsiveness and considers the throughput for the same symmetric algorithms, in megabytes per seconds (MB/s), to process a single file of 5MB, in a cycle of 500 iterations. The best throughput is given by Salsa20. Interestingly, Blake has shown slower performance than SHA-512 for large amounts of data.

VI. ENCRYPTTED FILESYSTEM WITH SECURE DELETION

In order to protect the secrecy of data during its entire lifetime, encrypted file systems must provide not only ways to securely store, but also reliably delete data, in such a way that recovering them from physical medium is almost impossible. The rationale behind the proposed solution is the actual possibility of performing secure deletion of files from ordinary Android applications, in user mode, without administrative privileges or operating system customization.

A. General description of the proposed solution

The proposed solution handles two cases according to the place where the deleted (or about to be deleted) file is stored:

1. The file is kept by the encrypted file system;
2. The file is logically deleted by the O.S.

1) Secure Deletion of Encrypted Files

The simplest way to fulfill the task of securely delete a file from an encrypted file system is to simply lose the encryption key of that file and then logically remove the file. This method does not need memory cleaning (purging) and is very fast. A prototype was built upon an Android port [44] for the EncFS encrypted file system [83]. Figure 9 illustrates the general behavior and functioning of the encrypted file system and its management application, called CryptoFM. The figure shows CryptoFM usage:

1. Inside CryptoFM, user sees a file system;
2. Inside, file names are decrypted on-the-fly;
3. Outside CryptoFM, user sees encrypted folders;
4. Inside, all file names are encrypted as well;
5. Outside, the file type is hidden;
6. Inside, corruptions are detected and monitored.

To accomplish the task of secure file deletion, the way EncFS manages cryptographic keys had to be modified. EncFS encrypts all files with a single master key derived from a password based encryption (PBE) function. It seems quite obvious that it is not feasible to change a master key

and encrypt the whole file system every time a single file is deleted. On the other hand, if each file were encrypted with its own key, then that key could be easily thrown away, turning the deleted file irrecoverable.

The modification to EncFS consists of the following:

a) Use PBE to derive a master key MK;
b) Use a Key Derivation Function (KDF) to derive a File System Encryption Key (FSEK) from MK;
c) Use an ordinary key generation function (e.g., PRNG) to generate a File Encryption Key (FEK);
d) Encrypt files along with their names using FEK and encrypts FEK with FSEK and random IV;
e) Keep a mapping mechanism from FEK and IV to encrypted file (FEK||IV → file).

A simple way to keep that mapping is to have a table file stored in user space as application's data. Care must be taken to avoid accidentally or purposely remove that file when cleaning device's user space. In Android devices, this can be done by rewriting the default activity responsible for deleting application's data. An application-specific delete activity would provide a selective deletion of application's data or deny any deletion at all. The removal from table of the FEK and IV makes a file irrecoverable. The ordinary delete operation then return storage space of that file to operating system. Figure 10 depicts the solution.

Another way to keep track of keys and files is to store the pair {FEK, IV} inside the encrypted name of the encrypted file. In this situation, a file has to be renamed before removed from the encrypted file system. The rename operation destroys the FEK and makes file irrecoverable. The ordinary delete operation then return storage space to operating system.

It is interesting to note that the proposed solution contributes to solve some known security issues of EncFS [84][85]. By using distinct keys for every file, a Chosen Ciphertext Attack (CCA) against the master key is inhibited. Also, it reduces the impact of IV reuse across encrypted files. Finally, it eliminates the watermarking vulnerability, because a single file imported twice to EncFS will be encrypted with two distinct keys and IVs.

Finally, the key derivation function is based upon PBKDF2 standard [59], keys and IVs are both 256 bits, and the table for mapping the pair {key, IVs} to files is kept by an SQLite scheme accessible only by the application.

2) Secure deletion of ordinary files

In this context, a bunch of files were logically deleted by the operating system for the benefit of the user, but they left sensitive garbage in the memory. Traditional solutions for purging memory cells occupied by those files are innocuous, because there is no way to know, from user's point of view, where purging data will be written.

An instance of this situation occurs when a temporary file is left behind by an application and manually deleted. This temporary file may be a decrypted copy of an encrypted file kept by the encrypted file system. Temporary unencrypted copies of files are necessary in order to allow other applications to handle specific file types, e.g., images, documents, and spreadsheets.

Whether temporary files will or will not be imported back to the encrypted file system, they have to be securely removed anyway. A premise is that the files to be removed are not in use by any application. The secure deletion occurs

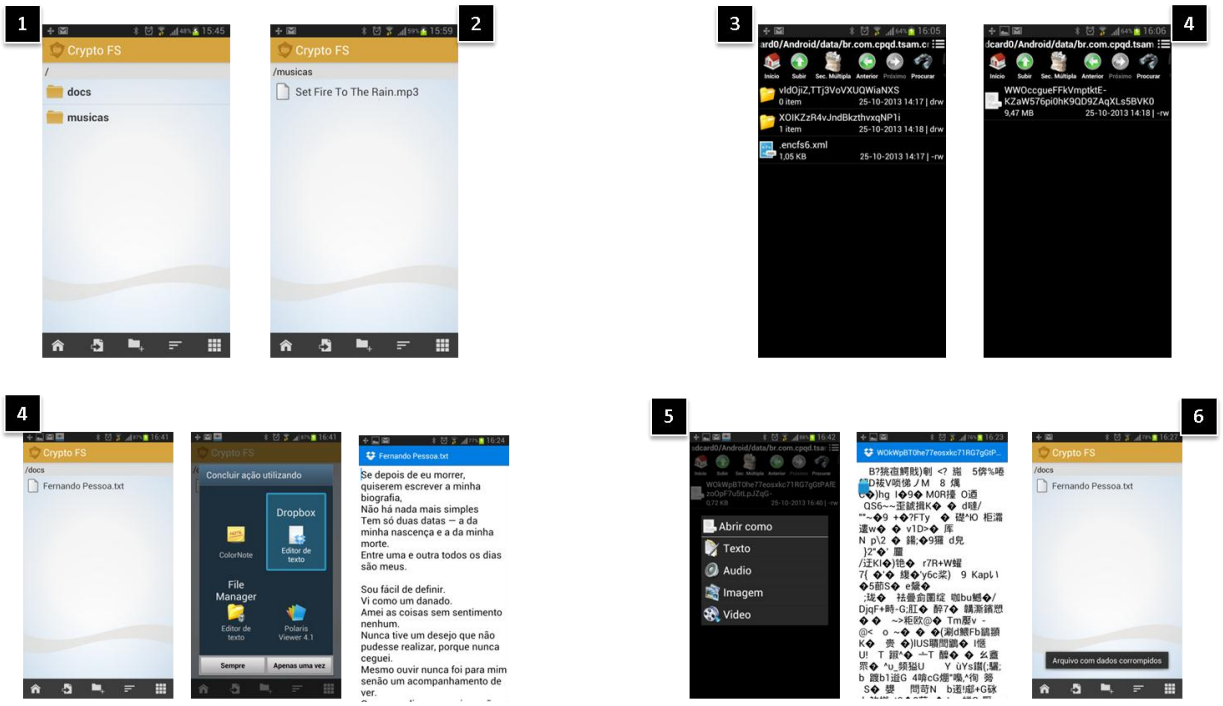


Figure 9. General behavior and functioning of the encrypted file system and its management application CryptoFM.

in three steps:

1) Logically remove targeted files with ordinary deletion;
2) Write a temporary file of randomized content that occupies all storage's free space;
3) When there is no free space anymore, logically delete that random file. That action purges all free storage in a way that no sensitive data is left behind.

The final result of this procedure is a flash storage free of sensitive garbage. Steps two and three can be encapsulated as a single function, called memory purging, and performed by an autonomous application. That application would be activated by the user whenever she needs to clean storage from sensitive garbage. The proposed solution adopted variations of this behavior.

Unfortunately, this procedure has two drawbacks. First, it takes time proportional to the size of the free space to be cleaned and the speed of memory writes. Second, this procedure, in the long term, if used with high frequency, has the potential to shorten the lifetime of flash memories.

In order to minimize the negative impact over memory life and avoid excessive delays during operation, steps two and three from above should not be carried out for every single file deleted from the system.

3) Limitations of the solution

The protection of cryptographic keys is of major importance. In spite of being stored encrypted, decrypted just before being used, and then released, the protection of cryptographic keys relies on Android security and the application confinement provided by that operating system. The proposed solution for memory purging is supposed to work in user-mode, as an ordinary mobile app, without administrative access, with no need for operating system modification, and using off-the-shelf devices. These decisions have consequences for security.

First of all, the solution is highly dependent on the way flash-based file systems and controllers behave. Briefly speaking, when the flash storage is updated, the file system writes a new copy of the changed data to a fresh memory block, remaps file pointers, and then erases the old memory blocks, if possible, but not certainly. This constrained design actually enables alternative implementations discussed further.

A second issue is that the solution is not specifically concerned about the type of physical memory (e.g., internal, external SD, NAND, and NOR) as long as it behaves like a flash-based file system. The consequence is that only software-based attacks are considered and physical attacks are out of scope.

Additionally, the use of random files is not supposed to have any effect on the purging assurance, but provides a kind of low-cost camouflage for cryptographic material (e.g., keys or parameters) accidentally stored on persistent media. An entropy analysis would not be able to easily distinguish specific random data as potential security material, because huge amounts of space would look random. Of course, this software-based camouflage cannot be the only way to prevent such attacks, but it adds to a defense in depth approach to security at almost no cost.

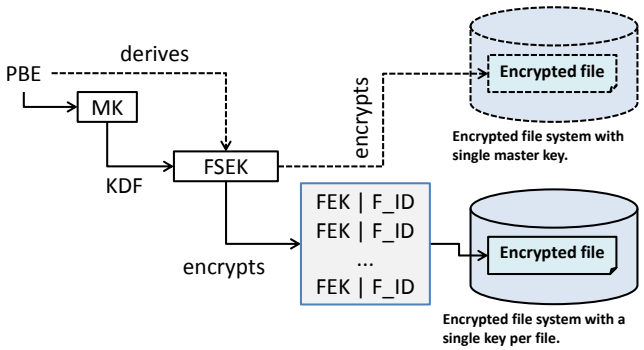


Figure 10. Extending an encrypted file system for secure deletion.

Finally, the purging technology described has passed all recovery tests performed with publicly available recovery tools, such as PhotoRec [86] and similar. That means, after purging, none of the recovery tools were able to recovery any deleted file. This confirms the feasibility of the purging technology for final users. On the other hand, advanced users may need deeper security assessments over physical hardware in order to trust the actual extend of the security provided by the proposed solution.

B. Alternative implementaions

The proposed solution for memory purging is a general policy for purging flash memories, and can be implemented in various ways, ranging from simple to complex implementations. In fact, a general solution has to offer different trade-offs among security requirements, memory life, and system responsiveness. The authors have identified three points for customization:

1. The period of execution for the purging procedure;
2. The size and quantity of random files;
3. The frequency of files creation/deletion.

Different trade-offs among the three customization points previously identified were implemented and evaluated. In all of them, the random file created in order to clean storage free space is called bubble, after the metaphor of soap cleaning bubbles over a dirty surface. These alternatives are discussed in next paragraphs.

1) Static single bubble

The simplest solution described in this text implements the idea of a single static bubble that increases in size until it reaches the limit of free space, and then bursts. This solution is adequate for the cases when storage has to be cleaned in the shortest period of time, with no interruption. A disadvantage is that other concurrent application can starve out of storage.

This solution is adequate when nothing else is happening, but the purging. Figure 11 illustrates, in four simple steps, the general behavior of this implementation:

1. Sensitive files are deleted logically;
2. The purging bubble is created and grows to occupy all available storage;
3. The bubble is logically removed when it reaches the limit of available storage;

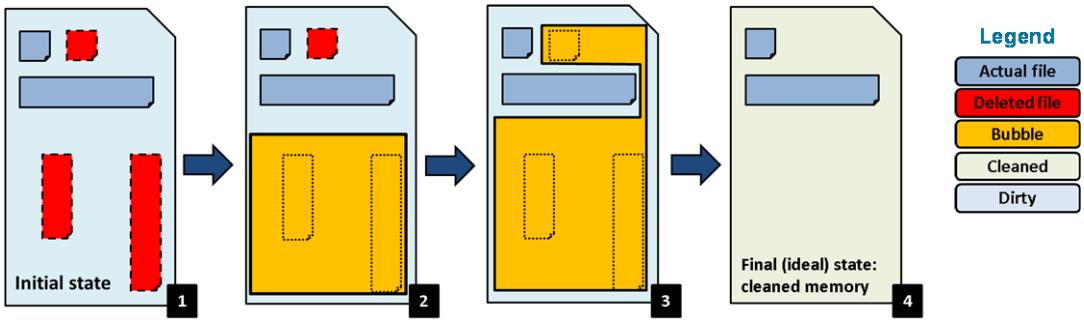


Figure 11. Purging strategy #1: Static Single Bubble.

4. The removed bubble leaves its waste, which overwrites any sensitive waste previously left in storage.

In that figure, an actual file is shown in blue, logically deleted files are in red, the bubble are in orange, dirty memory is light blue and purged memory is light grey.

2) Moving or sliding (single) bubble

In this alternative, a single bubble periodically moves itself or slides from one place to another. The moving bubble has size of a fraction of free space. For example, if bubble size is n^{-1} of free space, the moving bubble covers all free storage after n moves, considering the amount of free space does not change. A move is simply the rewriting of the bubble file, since flash memories will perform a rewrite in a different place. Figure 12 illustrates, in five simple steps, the general behavior of this implementation:

1. Sensitive files are deleted logically;
2. The purging bubble is created with a fraction of the available storage;

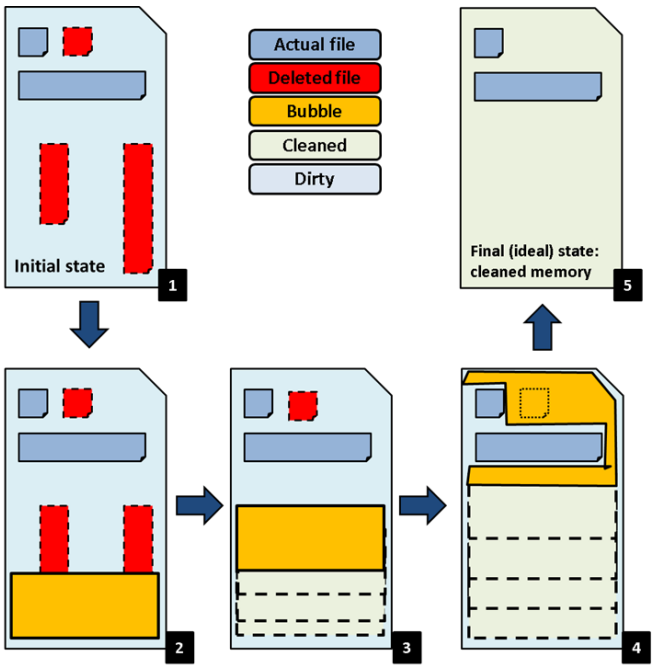


Figure 12. Purging strategy #2: Sliding single bubble.

3. The purging bubble moves due to rewriting behavior;
4. The bubble is logically removed when it has covered all the free space and have reached the limit;
5. The removed bubble leaves its waste, which overwrites any sensitive waste previously left in storage.

In a period of time equals to (T*(n/2)), where T is the time between moves, the chance of finding sensitive garbage in memory is 50%. This solution is adequate when storage has a low to moderate usage by concurrent applications. This solution preserves system responsiveness (usability) but diminishes security.

3) Moving or sliding (multiple) bubbles

This alternative uses more than one bubble instead of a single one. The size and amount of bubbles are fixed. For instance, if bubble size is n^{-1} of free space, two moving bubble covers all free storage space after n/2 moves each. The advantage of this method is to potentially accelerate memory coverage, reducing opportunity for memory compromising. Figure 13 illustrates the general behavior of this implementation:

1. Sensitive files are deleted logically;
2. The purging bubbles are created with a fraction of the available storage;
3. The bubbles move due to rewriting behavior;
4. Removed bubbles leave their wastes, which overwrite any sensitive waste previously left in storage.

In the example, two bubbles of size 1/n each can move at every T/2 period, and then concluding in (T*n). Alternatively, they can move at period T and terminate in 2*T*n, and so on. This solution is adequate when storage has a moderate usage by concurrent applications. This solution is probabilistic in the sense that as smaller the duration of T and greater the size of bubbles, greater the chance of successfully clean all memory.

4) Sparkling bubbles

This solution varies the size and amount of bubbles. The idea is to create a bunch of mini bubbles that are sparkled over free storage space. Bubbles are created and instantly removed at period T, which can be constant or random between zero and T. The sparking of bubbles stops when the sum of sizes for all created bubbles surpasses free space.

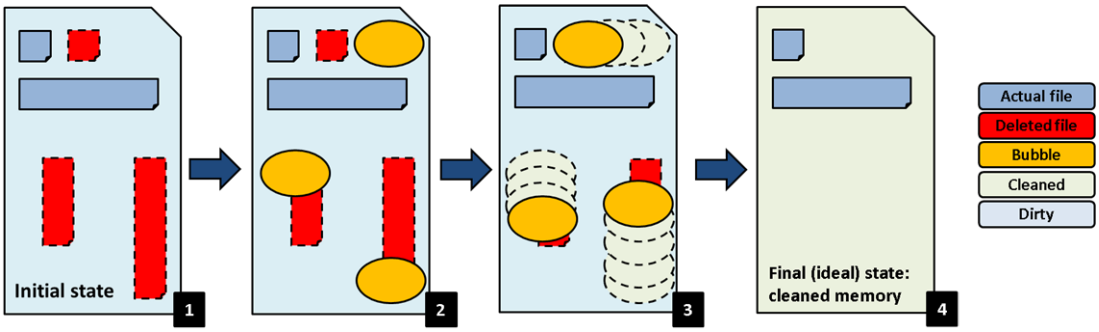


Figure 13. Purging strategy #3: Sliding (multiple) bubbles.

Bubble size can be small enough to not affect other applications. Figure 14 illustrates the general behavior of this implementation:

1. Sensitive files are deleted logically;
2. The bubbles are created with random size between a (specified) minimum and maximum;
3. The bubbles are removed and recreated concurrently;
4. The bubbles stop being created when the sum of their sizes reaches the size of free space;
5. Removed bubbles leave their wastes, which overwrite any sensitive waste previously left in storage.

This solution is adequate when storage has a moderate to high usage by concurrent applications. This solution is probabilistic in the sense that as smaller the duration of T, greater the chance of successfully clean the whole memory.

C. *Performance evaluation*

The four alternative implementations were compared according to their throughput for memory cleaning. That means, the rate at which data are purged, in gigabytes per minute (GB/min). This measure of purging speed tends to be more useful to compare storages of different size, such as internal and external memory. Performance tests were performed in two smartphones of type Motorola Atrix MB860, with Android 2.3.6 operating system, dual core 1GHz processor, 1GB of RAM and 16GB of internal storage (only 11 GB available to the end user). It was also used an SD Card (Class C) of 2GB. Random files created for purging had size of at most 2 GB or one tenth of free space. Performance measures were carried out in three scenarios:

- Scenario 1: mostly empty storage (~ 0-19%);
- Scenario 2: partially occupied storage (~ 20-80%);
- Scenario 3: mostly occupied storage (~ 81-99%).

In each scenario, both the internal and the external storage (SD card) were covered. Performance comparisons are structured as follows. First, a comparison is made between purging strategies for each occupancy scenario. Then, comparison is made between different occupancy scenarios for a specific strategy.

The implementations of the four purging strategies used concurrent threads if needed. The implementations of single static bubble and single sliding bubble used a single thread. The implementation of multiple sliding bubbles used two threads. The implementation of the mini-random bubbles used a minimum of five threads and at most twenty threads.

1) *Scenario 1 – mostly empty storage*

Performance measures for this scenario are shown in Figure 15(A). The storages were empty (0% occupancy). For this scenario, the following observations can be made:

a) The second purging strategy (single sliding bubble) is the fastest one. Apparently, this is because rewrite a single smaller file size is more efficient than continuously increase the size of a huge file;
b) The strategies with multiple bubbles are slower than the strategies with a single bubble. Probably, this is due to the overhead of managing multiple threads.
c) The higher the number of threads, worse the overall performance of purging, in slower CPUs. However, multi-bubble strategies are not blocking;
d) Purging of SD card was consistently slower in all cases.

Finally, data suggest that, when there is no competition for storage and storage is almost empty, sliding single bubble is the strategy that offers the best throughput. In situations

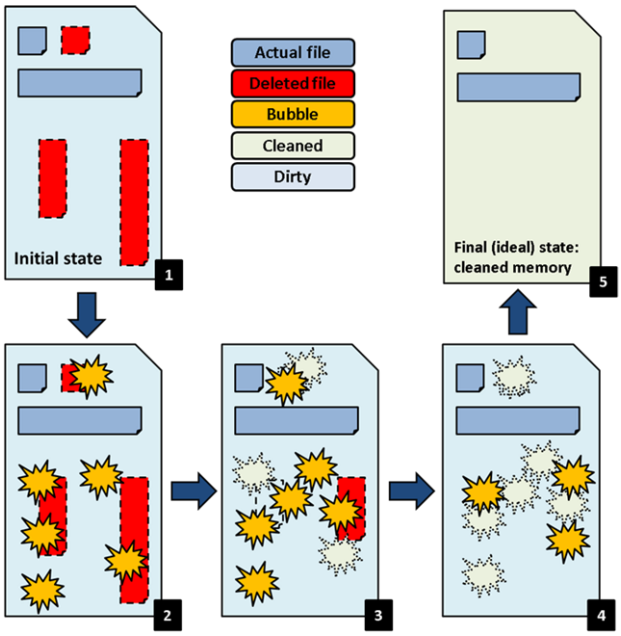


Figure 14. Purging strategy #4: Sparking bubbles.

where there is high competition for access to internal storage, multiple sliding bubbles seem to be more appropriate than the mini random bubbles.

2) Scenario 2 – partially occupied storage

Performance measures for this scenario are shown in Figure 15(B). The storages were partially occupied (30% occupancy). The following observations can be made:

a) The second strategy (single sliding bubble) is still the fastest one;
b) The strategies with multiple bubbles are slower than the strategies with a single bubble;
c) The higher the thread count, the worse the overall performance of purging;
d) Purging of SD card was consistently slower in all cases.

Finally, data suggest that, when there is no competition for storage and it is partially occupied, the single sliding bubble is still the strategy that offers the best throughput. However, single static bubble is very competitive. In situations where the competition for the internal storage is high, the throughputs for multiple sliding bubbles and mini random bubbles are quite similar.

3) Scenario 3 – mostly occupied storage

Performance measurements for this scenario are shown in Figure 15(C). The storages were nearly full (94% occupancy). The following observations can be made:

a) The first strategy (single static bubble) is just slightly faster than the second one (single sliding bubble);
b) The throughput of the strategies with multiple bubbles is close to the throughput of strategies with a single bubble, but showing a slightly worse performance;
c) The overall performance is still slightly worse with the increase number of threads;
d) Purging of SD card was consistently slower.

Data suggest that, when there is no competition for storage and its occupation is close to full capacity, the single bubble strategies (static or sliding) offer the best throughput.

4) Comparisom among strategies

Figure 16 compares all four strategies in different scenarios of occupancy (0%, 30% and 94%). All amounts are in GB/min. The following observations can be made:

1. In Figure 16(A), throughput of the single static bubble strategy improves with increasing storage occupation;
2. In Figure 16(B), throughput of the single sliding bubble gets worse with increased storage occupancy. This may be due to the slower treatment of memory rewriting;
3. In Figure 16(C), throughput of multiple sliding bubbles improves with increasing storage occupation. The use of two threads compensates for the relative slowness of the bubble rewriting;
4. In Figure 16(D), throughput of multiple random bubbles improves with increasing storage occupation. The use of multiple threads is combined with rapid generation of small bubbles.

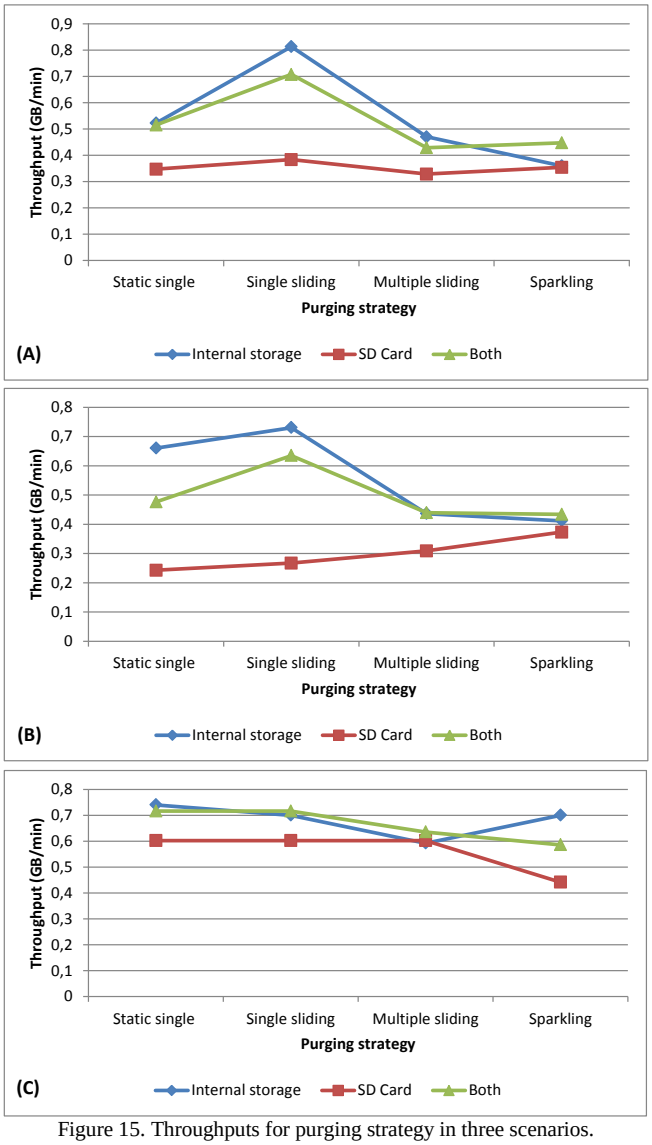


Figure 15. Throughputs for purging strategy in three scenarios.

The measurements show that the purging strategy with single sliding bubble has the highest throughput in average, being considered most appropriate in general. However, the static bubble is very competitive, though. In situations where there is high competition for internal storage, the throughput of strategies with multiple bubbles (sliding and random) is similar.

VII. Integrated view: Putting it All together

Among all technical challenges concerning the development of security features for applications on modern, Android-based, mobile devices, one of major importance is the integration of all these features into a security-ware framework. Figure 17 illustrates the high-level architecture of the proposed framework, where an

application container encapsulates all security features, including cryptography, key management, contact management, secure storage and deletion, access control, and mediated access to server-side applications. All these features are accessible to applications by means of APIs and services. Also, the framework promotes integration among mobile applications. For instance, the encrypted file system can be accessed by trusted applications inside the container.

Two main objectives drove the proposed architecture shown in Figure 17. The first one was to build a family of secure communication services over data packets (or over IP), through smartphones on public networks (e.g., 3G, 4G, Wi-Fi). The second was to develop tools for integrity checking and remote monitoring of smartphones, as well as techniques for active investigation on mobile platforms.

At the back office, the framework is supported by a laboratory for mobile security, which is able to carry out assessments on mobile environments, including platforms, applications and communications, as well as security analysis of mobile malware. The knowledge acquired by the lab team feeds the development team with security controls and counter measures. A private cloud provides services to the development team. Not only security services are provided, but also hosting for server-side applications.

VIII. CONCLUDING REMARKS

This paper discussed design and implementation issues on the construction of an integrated framework for securing both communication and storage of sensitive information over Android devices.

This text has shown how cryptographic services can be crafted to adequately fit secure communication services as well secure storage and deletion mechanisms, in such a way that security is kept transparent to the user, without being sacrificed. Also, a well-defined architecture allowed the selection and use of non-standard cryptography on a cryptographic library for Android.

The cryptographic library actually consists of both standard and non-standard cryptographic algorithms. Performance measurements were taken in order to compare cryptographic providers. Despite all difficulties to obtain realistic data, experiments have shown that standard cryptography can be competitive to other implementations. On the other hand, non-standard cryptography has shown low performance that can possibly limit its use in real time applications. However, their value consists in offering secure alternatives to possibly compromised standards. In fact,

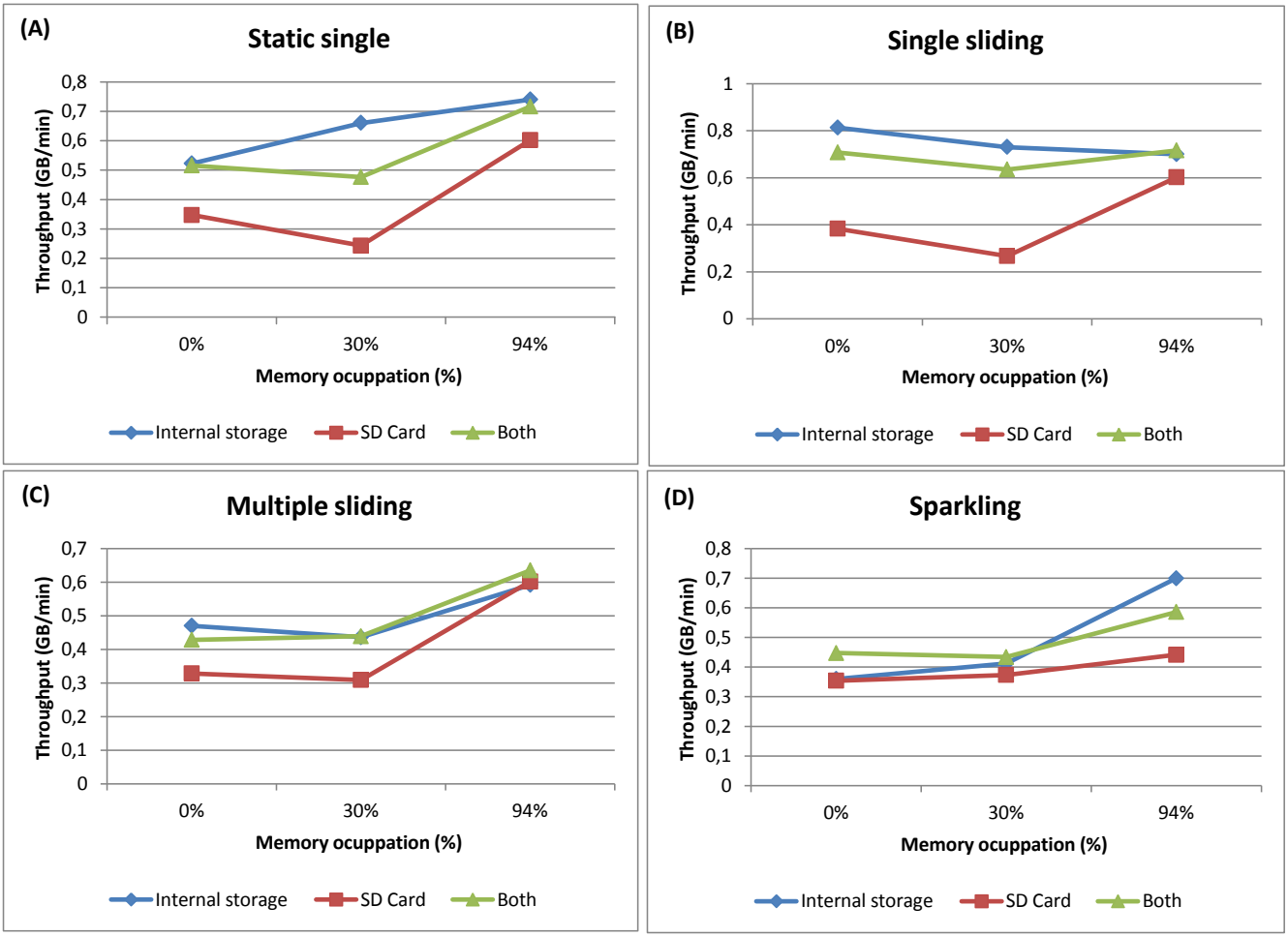


Figure 16. Throughputs by memory occupancy for purging strategy.

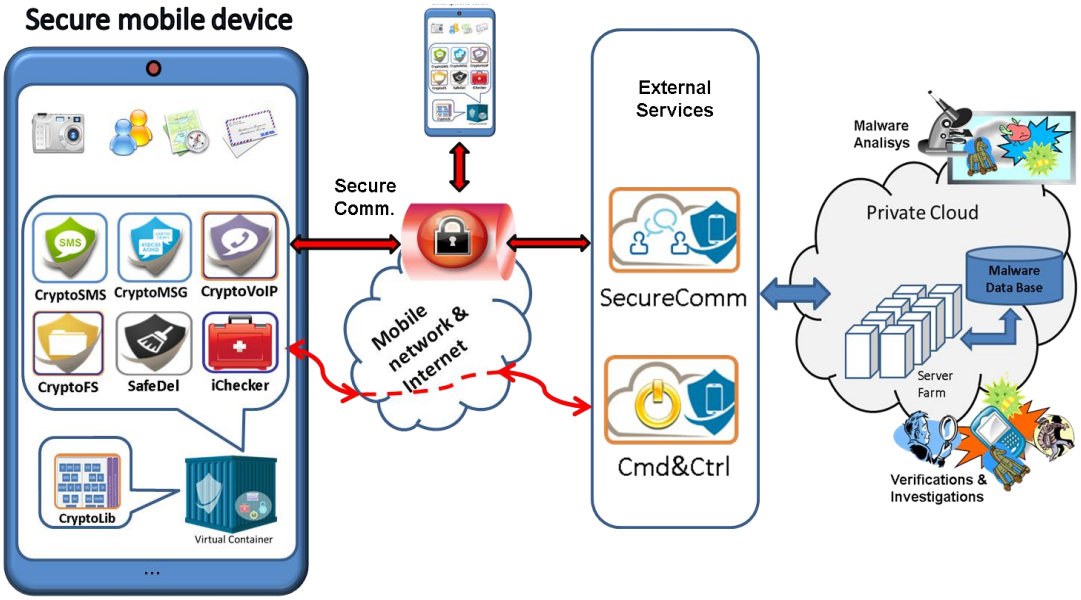


Figure 17. Framework high-level architecture with secure communication, trust management and back office services.

regarding recent global surveillance disclosures, non-standard cryptographic primitives can be faced as part of the usual trade-offs that directs the design of cryptographically secure applications.

Finally, the paper discussed the implementation of two user-level approaches to perform secure deletion of files. One works on secure deletion of encrypted files and the other handles deletion assurance of ordinary (unencrypted) files. Secure deletion of encrypted files was fully integrated to an encrypted file system and is transparent to the user. Secure deletion of ordinary files was fulfilled by an autonomous application activated under the discretion of the user. Performance measurements have shown that the approach is feasible and offers interesting trade-offs between time and deletion assurance.

In the short term, future work comprises the inclusion of additional secure applications to the mobile security framework, such as SMS, email, voice mail and VoIP. In the long run, the framework should evolve to a mobile platform for remote monitoring and fine-grained control of secure devices. Finally, as secure computing platforms become common place in mobile devices, the framework should be integrated to such features and provide strong, hardware-based protection to cryptographic keys.

ACKNOWLEDGMENT

The authors would like to acknowledge professor Julio Lopez, from University of Campinas, for his helpful comments. The authors acknowledge the financial support given to this work, under the project "Security Technologies for Mobile Environments – TSAM", granted by the Fund for Technological Development of Telecommunications – FUNTTEL – of the Brazilian Ministry of Communications, through Agreement Nr. 01.11.0028.00 with the Financier of Studies and Projects - FINEP/MCTI.

REFERENCES

[1] A. M. Braga and A. H. G. Colito, “Adding Secure Deletion to an Encrypted File System on Android Smartphones,” The Eighth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), 2014, pp. 106-110.

[2] A. M. Braga and D. C. Schwab, “Design Issues in the Construction of a Cryptographically Secure Instant Message Service for Android Smartphones,” The Eighth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), 2014, pp. 7-13.

[3] A. M. Braga and E. M. Morais, “Implementation Issues in the Construction of Standard and Non-Standard Cryptography on Android Devices,” The Eighth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), 2014, pp. 144-150.

[4] A. M. Braga, “Integrated Technologies for Communication Security on Mobile Devices,” The Third International Conference on Mobile Services, Resources, and Users (Mobility), 2013, pp. 47–51.

[5] A. M. Braga, E. N. Nascimento, and L. R. Palma, “Presenting the Brazilian Project TSAM – Security Technologies for Mobile Environments,” Proceeding of the 4th International Conference in Security and Privacy in Mobile Information and Communication Systems (MobiSec 2012). LNICST, vol. 107, 2012, pp. 53-54.

[6] A. Braga and E. Nascimento, “Portability evaluation of cryptographic libraries on android smartphones,” In Proceedings of the 4th international conference on Cyberspace Safety and Security (CSS'12), Yang Xiang, Javier Lopez, C.-C. Jay Kuo, and Wanlei Zhou (Eds.), Springer-Verlag, Berlin, Heidelberg, 2012, pp. 459-469.

[7] J. Menn, “Experts report potential software ‘back doors’ in U.S. standards,” retrived [May 2015] from http://www.reuters.com/article /2014/07/15/usa-nsa-software-idUSL2N0PP2BM20140715?irpc=932.

[8] NIST Removes Cryptography Algorithm from Random Number Generator Recommendations. Retrieved [May 2015] from http://www.nist.gov/itl/csd/sp800-90-042114.cfm.

[9] Java Cryptography Architecture (JCA) Reference Guide. Retrieved [May 2015] from docs.oracle.com/javase/7/docs/technotes/guides/ security/crypto/CryptoSpec.html.

[10] Java Cryptography Extension Unlimited Strength Jurisdiction Policy Files 7 Download. Retrieved [May 2015] from www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html.

[11] Java Cryptography Architecture (JCA) Reference Guide. Retrieved [May 2015] from docs.oracle.com/javase/7/docs/technotes/guides/security/crypto/CryptoSpec.html.

[12] The Java HotSpot Performance Engine Architecture. Retrived [May 2015] from www.oracle.com/technetwork/java/whitepaper-135217.html.

[13] Tuning Garbage Collection with the 5.0 Java Virtual Machine. Retrived [May 2015] from http://www.oracle.com/technetwork/java/gc-tuning-5-138395.html.

[14] Ergonomics in the 5.0 Java Virtual Machine. Available in: http://www.oracle.com/technetwork/java/ergo5-140223.html.

[15] A. Lux and A. Starostin, "A tool for static detection of timing channels in Java," Journal of Cryptographic Engineering, vol. 1, no. 4, Oct. 2011, pp. 303–313.

[16] D. Bornstain, "Dalvik VM Internals," retrieved [May 2015] from sites.google.com/ site/io/dalvik-vm-internals.

[17] H. Oh, B. Kim, H. Choi, and S. Moon, "Evaluation of Android Dalvik virtual machine," In Proceedings of the 10th International Workshop on Java Technologies for Real-time and Embedded Systems (JTRES '12), ACM, New York, NY, USA, 2012, pp. 115-124.

[18] W. Enck, D. Octeau, P. McDaniel, and S. Chaudhuri, "A study of Android application security," in Proceedings of the 20th USENIX conference on Security (SEC'11), USENIX Association, Berkeley, CA, USA, 2011, p. 21.

[19] M. Egele, D. Brumley, Y. Fratantonio, and C. Kruegel, "An empirical study of cryptographic misuse in android applications," in Proceedings of the 2013 ACM SIGSAC conference on Computer and Communications Security (CCS '13), 2013, pp. 73–84.

[20] P. Gutmann, "Lessons Learned in Implementing and Deploying Crypto Software," Usenix Security Symposium, 2002.

[21] NIST SP 800-38A. Recommendation for Block Cipher Modes of Operation. 2001. Retrieved [May 2015] from csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf.

[22] NIST SP 800-38D. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. 2007. csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf.

[23] P. Gutmann, "Secure deletion of data from magnetic and solid-state memory," proceedings of the Sixth USENIX Security Symposium, San Jose, CA, vol. 14, 1996.

[24] NSA (2012). Enterprise Mobility Architecture for Secure Voice over Internet Protocol. Mobility Capability Package - Secure VoIP, V 1.2.

[25] A. Voyiatzis, K. G. Stefanidis, and D. N. Serpanos, "Increasing lifetime of cryptographic keys on smartphone platforms with the controlled randomness protocol," in Proceeding of the Workshop on Embedded Systems Security (WESS'11), New York, NY, USA, 2011.

[26] J. Grosschadl and D. Page, "Efficient Java Implementation of Elliptic Curve Cryptography for J2ME-Enabled Mobile Devices," Cryptology ePrint Archive, Report Nr. 2011/712, 2011.

[27] M. Smith, C. Schridde, B. Agel, and B. Freisleben, "Secure mobile communication via identity-based cryptography and server-aided computations," J. Supercomput, vol. 55, no. 2, Feb. 2011, pp. 284-306.

[28] A. De Caro and V. Iovino, " jPBC: Java pairing based cryptography," In Proceedings of the IEEE Symposium on Computers and Communications (ISCC '11). IEEE Computer Society, 2011.

[29] ENISA, "Algorithms, key size and parameters report," nov. 2014. Retrived [May 2015] from www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/algorithms-key-size-and-parameters-report-2014.

[30] ENISA. "Study on cryptographic protocols," nov. 2014. Retrived [May 2015] from https://www.enisa.europa.eu/activities/identity-and-trust/library/deliverables/study-on-cryptographic-protocols.

[31] B. Xuefu and Y. Ming, "Design and Implementation of Web Instant Message System Based on XMPP," Proc. 3rd International Conference on Software Engineering and Service Science (ICSESS), Jun. 2012, pp. 83-88.

[32] D. T. Massandy and I. R. Munir, "Secured Video Streaming Development on Smartphones with Android Platform," Proc. 7th International Conference on Telecommunication Systems, Services, and Applications (TSSA), Oct. 2012, pp. 339-344.

[33] S. Schrittwieser et al., "Guess Who's Texting You? Evaluating the Security of Smartphone Messaging Applications," in Proc. 19th Network & Distributed System Security Symposium, Feb. 2012.

[34] Off-the-Record Messaging webpage. Retrieved [May 2015] from otr.cypherpunks.ca.

[35] H. Krawczyk, "SIGMA: "The 'SIGn-and-MAc' approach to authenticated Diffie-Hellman and its use in the IKE protocols," Advances in Cryptology-CRYPTO 2003, Springer Berlin Heidelberg, 2003, pp. 400-425.

[36] W. Diffie and M. Hellman, "New Directions in Cryptography," IEEE Transact. on Inform. Theory, vol. 22, no. 6, Nov. 1976, pp. 644-654.

[37] B. O'Higgins, W. Diffie, L. Strawczynski, and R. do Hoog, "Encryption and ISDN - A Natural Fit," International Switching Symposium (ISS87), 1987.

[38] W. Diffie, P. C. van Oorschot, and M. J. Wiener, "Authentication and Authenticated Key Exchanges," Designs, Codes and Cryptography (Kluwer Academic Publishers) 2 (2), 1992, pp. 107–125.

[39] Piercing Through WhatsApp's Encryption. Retrieved [May 2015] from blog.thijsalkema.de/blog/2013/10/08/piercing-through-whatsapp-s-encryption.

[40] A. Greenberg, "Whatsapp just switched on end-to-end encryption for hundreds of millions of users," Retrieved [May 2015] from www.wired.com/2014/11/whatsapp-encrypted-messaging.

[41] D. Boneh and R. J. Lipton, "A Revocable Backup System," in USENIX Security, 1996, pp. 91-96.

[42] K. Sun, J. Choi, D. Lee, and S.H. Noh, "Models and Design of an Adaptive Hybrid Scheme for Secure Deletion of Data in Consumer Electronics," IEEE Transactions on Consumer Electronics, vol. 54, no. 1, Feb. 2008, pp.100-104.

[43] S. M. Diesburg and A. I. A. Wang, "A survey of confidential data storage and deletion methods," ACM Computing Surveys (CSUR), vol. 43, no.1, p.2, 2010.

[44] Z. Wang, R. Murmuria, and A. Stavrou, "Implementing and optimizing an encryption filesystem on android," in IEEE 13th International Conference on Mobile Data Management (MDM), 2012, pp. 52-62.

[45] J. Reardon, S. Capkun, and D. Basin, "Data node encrypted file system: Efficient secure deletion for flash memory," in USENIX Security Symposium, 2012, pp. 333-348.

[46] J. Reardon, C. Marforio, S. Capkun, and D. Basin, "User-level secure deletion on log-structured file systems," in Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, 2012, pp. 63-64.

[47] J. Reardon, D. Basin, and S. Capkun, "On Secure Data Deletion," Security & Privacy, IEEE , vol. 12, no. 3, May-June 2014, pp.37-44.

[48] J. Reardon, D. Basin, and S. Capkun, "Sok: Secure data deletion," in IEEE Symposium on Security and Privacy, 2013, pp. 301-315.

[49] J. Reardon, H. Ritzdorf, D. Basin, and S. Capkun, "Secure data deletion from persistent media," in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security (CCS '13). ACM, New York, NY, USA, 2013, pp. 271-284.

[50] A. Skillen and M. Mannan, "On Implementing Deniable Storage Encryption for Mobile Devices," in 20th Annual Network & Distributed System Security Symposium, February 2013, pp. 24-27.

[51] A. Skillen and M. Mannan, "Mobiflage: Deniable Storage Encryption for Mobile Devices," IEEE Transactions on Dependable and Secure Computing, vol. 11, no. 3, May-June 2014, pp.224-237.

[52] P. Saint-Andre, K. Smith, and R. Tronçon, "XMPP: The Definitive Guide - Building Real-Time Applications with Jabber Technologies," O'reilly, 2009.

[53] W. Mao, "Modern cryptography: theory and practice", PTR, 2004.

[54] EJBCA PKI CA. Retrieved [May 2015] from http://www.ejbca.org.

[55] S. Fahl, M. Harbach, and H. Perl, "Rethinking SSL development in an appified world," in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security - CCS '13 (2013), 2013, pp. 49–60.

[56] Java Cryptography Architecture Oracle Providers Documentation for Java Platform Standard Edition 7. Retrieved [May 2015] from docs.oracle.com/javase/7/docs/technotes/guides/security/SunProviders.html.

[57] Java Cryptography Architecture Standard Algorithm Name Documentation for Java Platform Standard Edition 7. Retrieved [May 2015] from docs.oracle.com/javase/7/docs/technotes/guides/security/StandardNames.html.

[58] How to Implement a Provider in the Java Cryptography Architecture. Retrieved [May 2015] from docs.oracle.com/javase/7/docs/technotes/guides/security/crypto/HowToImplAProvider.html.

[59] B. Kaliski, "PKCS #5: Password-Based Cryptography Specification," Version 2.0, RFC 2898. Retrieved [May 2015] from tools.ietf.org/html/rfc2898.

[60] J. Bos, D. Osvik, and D. Stefan, "Fast Implementations of AES on Various Platforms," 2009. Retrieved [May 2015] from eprint.iacr.org/2009/501.pdf.

[61] NIST FIPS-PUB-197. Announcing the ADVANCED ENCRYPTION STANDARD (AES). Federal Information Processing Standards Publication 197 November 26, 2001.

[62] T. St. Denis. "Cryptography for Developers," Syngress, 2007.

[63] P. Barreto, AES Public Domain Implementation in Java. Retrieved [May 2015] from www.larc.usp.br/~pbarreto/JAES.zip.

[64] NIST FIPS-PUB-186. Digital Signature Standard (DSS). Retrieved [May 2015] from csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf.

[65] NIST FIPS-PUB-180-4. Secure Hash Standard (SHS). March 2012. Retrieved [May 2015] from csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf.

[66] NIST FIPS-PUB-198. The Keyed-Hash Message Authentication Code (HMAC). Retrieved [May 2015] from csrc.nist.gov/publications/fips/fips198/fips-198a.pdf.

[67] D. Aranha and C. Gouvêa, RELIC Toolkit. Retrieved [May 2015] from code.google.com/p/relic-toolkit.

[68] D. J. Bernstein, M. Hamburg, A. Krasnova, and T. Lange, "Elligator: elliptic-curve points indistinguishable from uniform random strings," in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, CCS '13, New York, NY, USA, 2013, pp. 967–980.

[69] D. Hankerson, A. J. Menezes, and S. Vanstone. "Guide to Elliptic Curve Cryptography," Springer-Verlag, New York, Inc., Secaucus, NJ, USA, 2003.

[70] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and Bo-YinYang, "High-speed high-security signatures," Journal of Cryptographic Engineering, vol. 2, no. 2, pp. 77–89, 2012.

[71] NIST FIPS PUB 186-2. Digital Signature Standard (DSS). Retrieved [May 2015] from csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf.

[72] R. Sakai, K. Ohgishi, and M. Kasahara. "Cryptosystems based on pairing," in Proceedings of the 2000 Symposium on Cryptography and Information Security (SCIS 2000), Okinawa, Japan, January 2000, pp. 26–28.

[73] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the Weil pairing," J. Cryptology, 17(4), Sept. 2004, pp. 297–319.

[74] F. Zhang, R. Safavi-Naini, and W. Susilo, "An Efficient Signature Scheme from Bilinear Pairings and Its Applications," in F. Bao, R. H. Deng and J. Zhou, ed., 'Public Key Cryptography', 2004, pp. 277-290.

[75] SHA-3 proposal BLAKE webpage. Retrieved [May 2015] from https://131002.net/blake.

[76] J. D. Bernstein, "The Salsa20 family of stream ciphers," Retrieved [May 2015] from cr.yp.to/papers.html#salsafamily.

[77] SERPENT webpage, "SERPENT A Candidate Block Cipher for the Advanced Encryption Standard," retrieved [May 2015] from www.cl.cam.ac.uk/~rja14/serpent.html.

[78] C. Petit, F. Standaert, O. Pereira, T. G. Malkin, and M. Yung, "A Block Cipher Based Pseudorandom Number Generator Secure Against Side-Channel Key Recovery," in Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS '08), 2008, pp. 56–65.

[79] G. Anthes, "French team invents faster code-breaking algorithm," Communications of the ACM, vol. 57, no.1, January 2014, pp. 21-23.

[80] R. Barbulescu, P. Gaudrey, A. Joux, and E. Thomé, "A quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic," June 2013, preprint available at http://eprint.iacr.org/2013/400.pdf.

[81] The Legion of the Bouncy Castle webpage. Legion of the Bouncy Castle Java cryptography APIs. Retrieved [May 2015] from www.bouncycastle.org/java.html.

[82] SpongyCastle webpage, Spongy Castle: Repackage of Bouncy Castle for Android, Bouncy Castle Project (2012), Retrieved [May 2015] from rtyley.github.com/spongycastle/.

[83] V. Gough, "EncFS Encrypted Filesystem," stable release 1.7.4 (2010). Retrived [May 2015] from http://www.arg0.net/encfs.

[84] M. Riser, "Multiple Vulnerabilities in EncFS," 2010. Retrieve [May 2015] from: http://archives.neohapsis.com/archives/fulldisclosure/2010-08/0316.html.

[85] T. Hornby, "EncFS Security Audit," retrived [May 2015] from: https://defuse.ca/audits/encfs.htm.

[86] PhotoRec, Digital Picture and File Recovery. Retrived [May 2015] from: http://www.cgsecurity.org/wiki/PhotoRec.

Symbolic Execution Based Automated Static Bug Detection for Eclipse CDT

Andreas Ibing

Chair for IT Security
TU München, Germany
Email: andreas.ibing@tum.de

***Abstract*—Software vulnerabilities may be exploited for intruding into a system by an attacker. One approach to mitigation is to automatically analyze software source code in order to find and remove software bugs before release. A method for context-sensitive static bug detection is symbolic execution. This article presents an SMT-constrained static symbolic execution engine with sound path merging. The engine is used by checkers for memory access violation, infinite loops, and atomicity violations. Context information provided by the engine is shared by the different checkers. Further checkers can easily be connected. The engine integrates as plug-in extension into Eclipse CDT and uses CDT's parser, AST visitor and CFG builder, as well as Eclipse's GUI and marker framework for bug reporting. The presented approach is evaluated with test cases from the Juliet test suite for C/C++. The evaluation shows a significant speed-up by path merging already for the small Juliet programs. The speed-up depends on the number of decision nodes with more than one satisfiable branch and increases for larger programs.**

Keywords–Static analysis; Symbolic execution.

I. Introduction

This article is an extended version of [1], which presents a backtracking symbolic execution engine with sound path merging on the C source level. This extended version gives a more detailed description, provides more context information and evaluates the symbolic execution engine on a larger test set. Software weaknesses are classified by the common weakness enumeration [2]. For brevity, a software weakness is called bug in this article. If a weakness could be exploited by an attacker, it is a vulnerability. The likelihood of exploit varies for different bug types. For buffer overflows, e.g., the likelihood of exploit is very high [2].

Symbolic execution [3] is a program analysis method, where software input is regarded as variables (symbolic values). It is used to automatically explore different paths through software, and to compute path constraints as logical equations from the operations with the symbolic input. An automatic theorem prover (constraint solver) is used to check program paths for satisfiability and to check bug conditions for satisfiability. The current state of automatic theorem provers are Satisfiability Modulo Theories (SMT) solvers [4], the standard interface is the SMTlib [5]. An example state-of-the art solver is described in [6].

Automatic analysis tools that rely on symbolic execution have been developed for the source-code level, intermediate code and binaries (machine code). Available tools mostly analyze intermediate code, which exploits a small instruction set and certain independence of programming language and target processor. A prominent example is [7], which analyzes Low Level Virtual Machine (LLVM [8]) code. Symbolic execution on the source-code level is also interesting for several reasons. An intermediate representation loses source information by discarding high-level types and the compiler lowers language constructs and makes assumptions about the evaluation order. However, rich source and type information is needed to explain discovered bugs to the user [9] or to generate quick-fix proposals.

In order to detect bugs as early as possible, bug detection tools should be integrated into IDEs. The integration of bug finding tools into IDEs is further important for ease of use and for the integration of different tools. A synergy lies for example in the automated generation of quick-fix refactoring proposals based on detected bug information.

During symbolic execution, a symbolic execution engine builds and analyzes satisfiable paths through programs, where paths are lists of control flow graph (CFG) nodes. Always restarting symbolic execution from the program entry point for different, partly overlapping program paths (path replay) is obviously inefficient. The standard approach is therefore the worklist algorithm [10]. In this algorithm, a list of symbolic program states (the worklist) is kept in memory. These states are the frontier nodes (unexplored nodes) of the program execution tree. While the list is not empty, one symbolic program state is taken from the list and interpreted to yield its successor state(s), which are then added to the list. At program branches, there may be more than one satisfiable successor state. In this case the respective predecessor is cloned before interpretation. The reuse of intermediate analysis results with state cloning has the downside of being memory-intensive. In [11], state cloning with a recursive data structure to store only state differences is used. Another approach for engine implementation is symbolic state backtracking [12]. It keeps only the symbolic program states along the currently analyzed program path in memory (stored incrementally with single assignments) and avoids the inefficiency of path replay as well as the exponential memory consumption of state cloning.

The tree of satisfiable program paths, called the program execution tree, grows exponentially with the number of decisions in the program for which two or more branches are satisfiable. Straight-forward application of symbolic execution is therefore not scalable. This is often called the path explosion problem. In [13], it is noted that program paths can be merged when the path constraints differ only in dead variables, because further path extension would have the same consequences for the paths. It presents an implementation that extends [11]. This implementation uses a cache of observed symbolic program states and introduces a type of live variables analysis, which

it calls read-write-set (RWSet) analysis.

Interesting properties of bug detection algorithms are soundness (no false positive detections) and completeness (no false negatives). Because a bug checker cannot be sound and complete and have bounded runtime, in practice bug checkers are evalutated with measurement of false positive and false negative detections and corresponding runtimes on a sufficiently large bug test suite. The currently most comprehensive C/C++ bug test suite for static analyzers is the Juliet suite [14]. Among other common software weaknesses, it contains test cases for buffer overflows, infinite loops and race conditions. In order to systematically measure false positives and false negatives, it contains both 'good' and 'bad' functions, where 'bad' functions contain a bug. It further combines 'baseline' bugs with different data and control flow variants to cover the language's grammar constructs and to test the context depth of the analysis.

This paper develops and evaluates a sound path merging method in a source-level backtracking symbolic execution engine. The aim is to context-sensitively find bugs in unannotated C code, in the sense of automated testing without test-suite, while alleviating the path explosion problem. The approach is targeted at all C bug types that can be detected as constraint violations. The implementation extends [12]. According to the TIOBE index [15], C is currently the most popular programming language (based on average ranking during the last 12 months). The remainder of this paper is organized as follows. Section II gives an overview of related work. Section III shortly reviews symbolic execution. Section IV describes the tool architecture and design decisions. The description includes the integration in the Eclipse C/C++ development tools (CDT). Section V depicts different checker classes connected to the symbolic execution engine. These checkers are described in more detail in previous publications [12], [16], [17], [18]. Section VI presents results of experiments with test cases from the Juliet suite, with focus on the analysis speed-up provided by path merging. Section VII then discusses the presented approach based on the results.

II. Related Work

There is a large body of work on symbolic execution available, which spans over 30 years [19]. Therefore, only a small selection is named here. More in-depth information is available in survey articles [19], [20], [21], [22].

a) Related approaches: There are several approaches that are closely related to automated bug detection with symbolic execution. One approach is annotation-based verification, which proves the absence of errors. The annotations reduce the context that is necessary for analysis. An annotation language for C is presented in [23], one for Java in [24]. Prominent verification tools for C are described in [25], [26]. Another approach is symbolic model checking [27], where the whole program is treated as a formula. Bounded model checking for C is described in [28]. An approach that offers a smooth transition between static analysis and verification is extended static checking [29]. Symbolic execution can be seen as an instance of abstract interpretation, because some variables have formulas as values, which abstracts from concrete interpretation. Abstract interpretation in a narrower sense [30] is referred to in the paragraph on abstraction.

b) Different levels of software: Symbolic execution has been applied on the software architecture level to models, e.g., to UML-RT state diagrams [31]. Further related state-based work on the model level is presented in [32], [33]. On the source code level, symbolic execution has been applied to a variety of languages. Examples for C are [34], [11]. Most tools perform symbolic execution on an intermediate code representation. Apart from [7], where LLVM intermediate code is analyzed, prominent symbolic execution engines are presented in [35] and [36]. In [35], dynamic symbolic execution of the Common Intermediate Language (MSIL/CIL) is performed. The engine described in [36] analyzes Java bytecode. Binary code has been analyzed by lifting to an intermediate representation and symbolic execution of the intermediate code, described in [37], [38]. Analysis of x86 binaries with symbolic execution is presented in [39].

c) Static and dynamic: Symbolic execution can be applied both as static and as dynamic analysis. The latter is also referred to as concolic testing [40]. Dynamic symbolic execution for test case generation is described in [7], [11], [35], [39], [40], [41]. To reduce complexity and increase analysis speed, as many variables as possible are regarded as concrete values. Normally, only input variables or variables that directly depend on program input are modelled as symbolic. The analysis runs dynamically as long as all parameters are concrete, and equation systems for the solver are smaller. In [40], [41], dynamic symbolic execution is applied on the C source code level. In [39], dynamic symbolic execution is applied for the analysis of x86 machine code. A combination of static and dynamic symbolic execution called selective symbolic execution is presented in [42]. The approach is to specify a system part of interest where symbolic variables are used, and to execute other parts with concrete/concretized values. It uses a hypervisor with LLVM backend and utilizes the symbolic execution engine described in [7].

d) Path merging: Sound path merging based on dead path differences is presented in [13], the implementation extends [11]. Merging of paths with live differences is investigated in [43]. Path disjunctions are used in the corresponding logic formulation passed to the solver. Heuristics for path merging are presented, which aim at balancing computational effort between the symbolic execution frontend and the SMT solver backend. The implementation extends [7].

e) Abstraction: Other related work uses abstraction, i.e., generalization of constraints, to merge more paths. Abstract interpretation [30] allows for complete bug detection (no false negatives), but introduces false positives (unsound). An approach to automatically generate an abstraction based on predicates over decision conditions contained in the program source is presented in [44]. Counter-example guided abstraction refinement [45] is an automated abstraction refinement to iteratively undo unsound path merges in order to remove false positives. Another method is presented in [46] and further developed in [47], [48], [49]. It uses logic interpolation [50] and weakest precondition computing during backtracking of error-free paths, so that further error-free paths explored later could be merged.

f) IDE integration: This paragraph considers only the widely used open-source Eclipse IDE. It is possible to connect external tools as processes to the IDE. There are, e.g., Eclipse

plug-ins available that communicate over character streams with the analysis engines presented in [25], [28]. Eclipse is designed according to the OSGi architecture (formerly known as Open Services Gateway initiative) [51], i.e., it consists of a small runtime, and functionality is provided as plug-ins. It contains its own dependency and update management. Therefore, tight IDE integration offers several advantages, the most important one being the synergies between plug-ins. Eclipse CDT features a code analysis framework [52]. Eclipse also includes a SAT solver [53]. On the other hand, it currently (version 4.4) does not feature an SMT solver, and CDT's code analysis framework does not include path-sensitive or context-sensitive analyses.

III. Symbolic Execution

Symbolic execution can be seen as a state transition system (e.g., [48], [54]). This section shortly describes interprocedural symbolic execution for whole-program analysis.

A. Symbolic program state

A symbolic program state is a quadruple $\sigma = (l, s, \Pi, T)$. l is a program location. On the source code level this means a control flow node. To this program location corresponds a syntax subtree $a(l)$ of the abstract syntax tree (AST) of a source file. s denotes the set of symbolic program variables, i.e., logic equations over symbolic input variables. Π is the path constraint, and T the function call stack.

B. Successor locations

Depending on the type of l, there are the following cases for the set of successor locations of a symbolic program state:

1) $a(l)$ contains one or more not yet evaluated function call expressions. Then the successor location is the start node of the function whose function call expression is next in traversal order of $a(l)$. Location l is saved as return node: $T.push(l)$.
2) l is an exit node and $a(l)$ does not contain an unevaluated call expression. The successor location is the return node from the stack: $l' = T.pop()$. In case of exit from `main()`, symbolic execution terminates.
3) l is *not* an exit node and $a(l)$ does not contain an unevaluated call expression. Then, the set of successor locations are l's children along the edges in the function's control flow graph.

C. Symbolic successor states

For the transition from l to successor l' there are the following cases:

1) l' is a start node. Then $\sigma' = (l', s \wedge c_C, \Pi, T')$, where $c_C = [a_C(l)]$ is the evaluation of the subtree of $a(l)$ rooted in the respective function call expression.
2) l' is a branch node. Then $\sigma' = (l', s, \Pi \wedge c, T')$, where the constraint $c = [a(l)]$ is the evaluation of $a(l)$. That is, the branch condition is added to the path constraint.
3) l' is neither a start node nor a branch node. Then $\sigma' = (l', s \wedge c, \Pi, T')$, i.e., the constraint $c = [a(l)]$ is added for the symbolic variables. If l was an exit node, then the evaluation $c = [a(l)]$ continues with the return value for the respective function call expression.

IV. Architecture and Implementation

A. Main classes and Eclipse integration

The tool is a plug-in for the Eclipse IDE and extends the CDT code analysis framework (Codan [52]). Eclipse CDT provides a C/C++ parser and AST visitor, and the code analysis framework provides a control flow graph builder. Codan uses Eclipse's marker framework for reporting bugs in the CDT GUI.

The main classes of the architecture are illustrated in Figure 1. `WorkPoolManager` and `Worker` are active classes. The main functions of the classes are [12]:

- `WorkPoolManager` implements Codan's `IChecker` interface and through this becomes callable from the Eclipse GUI (through Codan). It starts `Worker` threads and reports found bugs through the Codan interface to the Eclipse marker framework. As synchronization object, `WorkPool` is used (synchronized methods). It tracks the number of active `Worker`s and serves for dynamic work re-distribution.
- `Worker` explores a part of the program execution tree specified by a start path, with help of its `Interpreter`. Different `Worker`-threads concurrently analyse disjunct partitions of a program's execution tree. `Worker` has a forward and a backward (backtracking) mode. It passes references to control flow graph nodes for entry (forward mode) or backtracking to its `Interpreter`.
- `Interpreter` performs symbolic interpretation according to the tree-based interpretation pattern [55]. The control flow node processor (`CFNodeProcessor`) implements CDT's AST visitor and translates into SMTlib logic equations.
- `SMTSolver` wraps the SMT solver. In the current implementation, the wrapper is configured to call the SMT solver described in [6].
- `IPathObserver` is an interface provided by the `Interpreter` for checker classes. Checkers can register for notifications and can access context information.
- `BranchValidator` checks branches for satisfiability with a solver query and throws an exception (caught by the `Worker`) in case of unsatisfiability, which causes pruning of the respective path. `BranchValidator` is triggered when entering a branch node.
- `ProgramStructureFacade` provides access to control flow graphs.

B. Tree-based interpretation

First, all source files of the program under test are parsed into ASTs, and a CFG is generated for each AST subtree

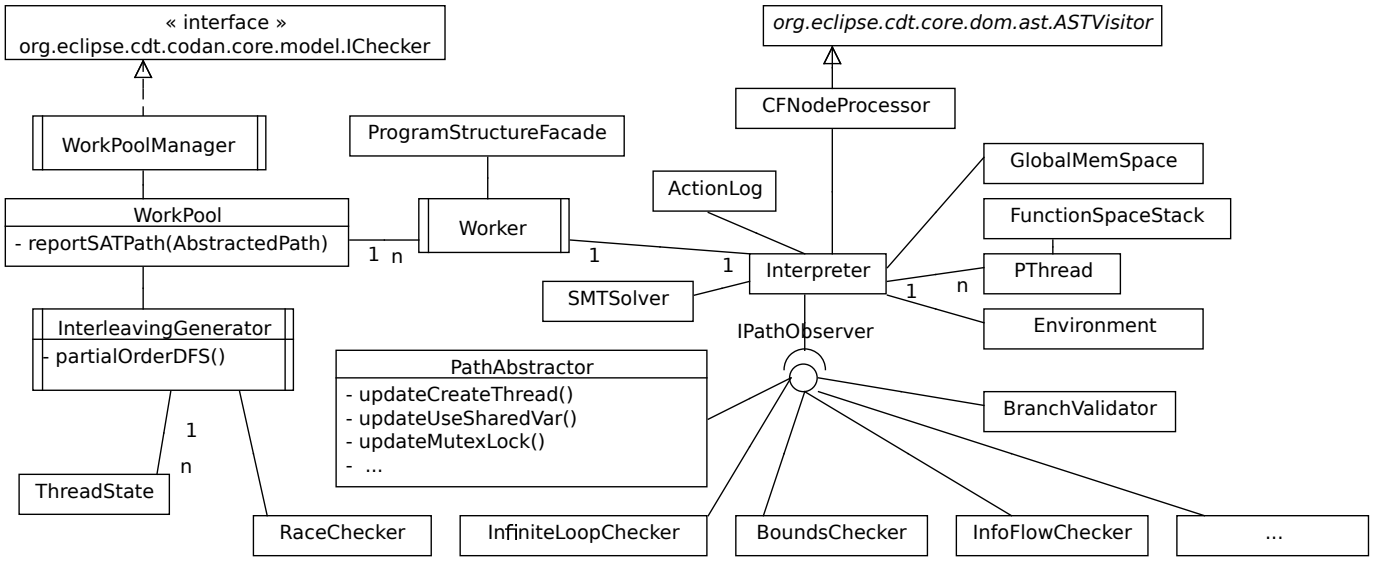


Figure 1. Architecture, main classes

that is rooted in a function definition. Symbolic execution traverses CFGs, beginning from `main()` start. For each control flow node, the corresponding AST subtree is interpreted. The maximum loop depth to be explored can be bounded. Symbolic variables are stored in and retrieved from a memory system, which consists of the classes `GlobalMemSpace` and `FunctionSpaceStack`. Symbolic variables are resolved by their syntax tree name (CDT's `IASTName`) and binding. Each `Interpreter` instance supports multiple thread objects (`PThread`) of which each has its own stack object (`FunctionSpaceStack`, compare Figure 1). Variable dependencies are traced. This is used for slicing, so that solver queries only contain the necessary subset of logic equations. The program under test communicates with its environment through the operating system API, which is wrapped by the C standard library. Symbolic execution is not extended into the standard library. Instead, symbolic function models can be provided for standard library functions (through the `Environment` class). In order to allow for backtracking of the symbolic program state, the semantic actions performed per CFG node, like, e.g., variable declarations, are stored in an `ActionLog`. The interpreter passes AST subtree references, which are referenced by CFG nodes, to `CFNodeProcessor` for translation.

Figure 2 illustrates the data structures for CFG and AST, which are provided by CDT/Codan. The figure shows on the left the control flow for an example function from [14]. On the right it shows two AST subtrees which are referenced by two control flow nodes. There are eight satisfiable paths through the function. One of these paths contains a buffer overflow bug. This path is depicted in red. All eight paths could be merged at the function exit. The function's exit node is depicted in blue.

In general, the interpretation works per CFG node. The current CFG node is interpreted, then a successor node is chosen, who is interpreted next. An exception are function calls. If a CFG node corresponds to a statement or expression that contains a function call, then the node is first only partially interpreted, i.e., up to the function call expression (which includes parameter collection). Then the called function's start node and respective successors are interpreted. After exit of the called function, interpretation continues with the remaining uninterpreted AST subtree part of the calling CFG node, with the function's return value.

C. Translation into SMTlib logic

Translation is implemented by bottom-up traversal of an AST subtree according to the visitor pattern [56]. This pattern is commonly used for operations on a graph of elements (here the AST), where the operation on a node depends on the node's runtime type. The class `CFNodeProcessor` implements CDT's `ASTVisitor` (compare Figure 1). Translation attributes are passed upwards during AST traversal. Attributes can be for example intermediate translation results. During translation, type promotion is performed according to the operators. The translation uses single assignments to avoid desctructive updates. Pointers and structs are not directly translated into SMT logic, they are represented internally during interpretation (e.g., a pointer has a target and an offset formula). Logic equations are generated at pointer dereference and at field access to a struct. The translation output are equation in the SMTlib sublogic of arrays, uninterpreted functions and nonlinear integer and real arithmetic (AUFNIRA). While the translation in general works per CFG node, one exception are function call expressions as mentioned in the last subsection. Another exception are `switch` statements – where the default branch's formula depends on all sibling CFG nodes.

D. Analysis of multi-threaded code

Analysis of multi-threaded code is supported for programs which use a subset of the Portable Operating System Interface (POSIX) threads API. Certain functions from the POSIX threads library like mutex locking and unlocking, and creation and joining of other threads are currently supported by function models [18]. The symbolic execution is run with a pre-defined

scheduling algorithm. The implementation uses 'lowest thread-id first' scheduling. A thread blocks (becomes inactive), e.g., by trying to acquire a lock already held by another thread or with a join call for a thread that is still alive. Unless the code contains race condition bugs, the program behaviour is identical for all possible schedulings, so one scheduling algorithm suffices.

If race conditions are also to be detected, a recording and abstraction of satisfiable paths can be activated. Scheduling can in principle occur between any assembler instructions. But most actions of different threads are independent and commutative. This can be exploited by a partial-order reduction [57]. Only thread interactions are relevant for race detection, where thread interactions are events like thread creation, joining, mutex locking and unlocking, and also the read or write access to shared variables. Whether or not a variable is shared between threads is in principle context-sensitive. It depends on the current program path including the current function's call context. All global variables are marked as shared when they are first accessed. Then the shared property is inferred over data flow constructs like assignments, references, function call parameters and return values. Further, the thread start arguments are also marked as shared. Shared variables are traced by the `Interpreter`, and all thread interaction events are recorded by the class `PathAbstractor`.

The analysis of multi-threaded code does not require more effort than the analysis of single-threaded code. Extra effort is only spent if race conditions are to be detected. Race condition analysis works on the recorded thread interaction abstraction level [18] and is described in more detail in Section V-D.

E. Multi-threaded engine

The implementation is multi-threaded, a configurable number of worker threads concurrently explores different parts of the execution tree [12]. Each worker performs a depth-first exploration of its partition with backtracking of the symbolic program state. Control flow graphs and syntax trees are shared between worker threads. AST nodes are not thread-safe. Workers therefore lock AST subtrees at the CFG node level, i.e., the AST subtree that is referenced by the currently interpreted CFG node. Dynamic redistribution of work between workers is enabled by splitting a workers partition of the execution tree at the partition's top decision node, where a partition is defined by the start path leading to its root control flow decision node. The concatenations of the partition start path and one of the branches not taken by the current worker are returned as start paths for other workers. After a split, the current worker's start path is also prolonged by one branch node, which is the branch node that the worker had taken. The current worker's prolonged start path still defines its execution tree partition, which is now reduced in size. Analysis starts with one worker, who splits its partition until the configured number of workers is busy. A worker is initialized by replaying its partition start path. If a worker reaches an unsatisfiable branch or a satisfiable leaf of the execution tree, it backtracks and changes a path decision according to depth-first tree traversal. If backtracking reaches the end of the partition start path, the partition is exhausted. The `WorkPool` is used for synchronization. It serves to exchange split paths between workers and tracks the number of active workers. A parallelization speed-up is possible if the program under test has decisions for which more than one branch is satisfiable [12].

F. Backtracking and path merging

1) Dead and live variables: Paths can be merged without any loss in bug detection accuracy when the path constraints and symbolic variable constraints differ only in dead variables [13]. The detection of such merge possibilities requires a context cache at potential merge points. Also required is a way to detect dead variables and to filter them from the path and variable constraints. Therefore, potentially interesting merge points are program locations where the sets of dead and live variables change. Such points are function start and function exit and after scope blocks like `if` / `else` or `switch` statements and loops.

2) Merge points: Path merges are performed at function exit in the current implementation. Merges are possible because stack frame variables die at function exit. Path and variable constraints at function exit are treated as concatenation of the function's call context and the local context. The approach misses possibilities to merge paths earlier after scope blocks inside one function. On the other hand, it does not require more complex live variable analysis at intermediate points. The approach merges paths that have split inside the same function, possibly with other function calls in between. It needs to know the set of variables that have been written since the merge paths have split. This is overapproximated by the set of variables written since entering the function that is left at the program location in question. A set of potentially read variables along path extensions is not computed. From the set of variables that have been written as local context (i.e., since function entry), global variables, the return value and all variables that have been written through pointers (pointer escape, potential write to, e.g., other stack frame) are assumed as live. The remaining written local variables are assumed as dead, which is a sound assumption. The local context is then reduced by removing the dead variables. A context cache is used to lookup observed reduced local contexts from pairs of a function's exit node (in the function's control flow graph) and call context. During symbolic execution, at each exit node the context cache is queried for a merge possibility. Then the current path is merged if possible, otherwise the local reduced context is added as new entry to the context cache.

3) Backtracking: Due to single assignment form, a symbolic program state contains all previous states along the path. Backtracking is enabled by class `ActionLog`, which records certain semantic actions performed for CFG nodes on the current path (e.g., variable creation or hiding). For example, if a function exit is backtracked, the function's stack frame with contained variables must be made visible again. Dead variables are therefore not garbage-collected, because this would impede backtracking. The engine further allows to record and visualize explored parts of a program execution tree.

4) Path merging: Path merging needs knowledge about the sets of written variables since path split. The implementation uses the class `ActionLog` to derive this information. It contains all writes to variables, including writes to globals and writes through pointers (potentially to other stack frames). The action log is looked through backwards up to the current

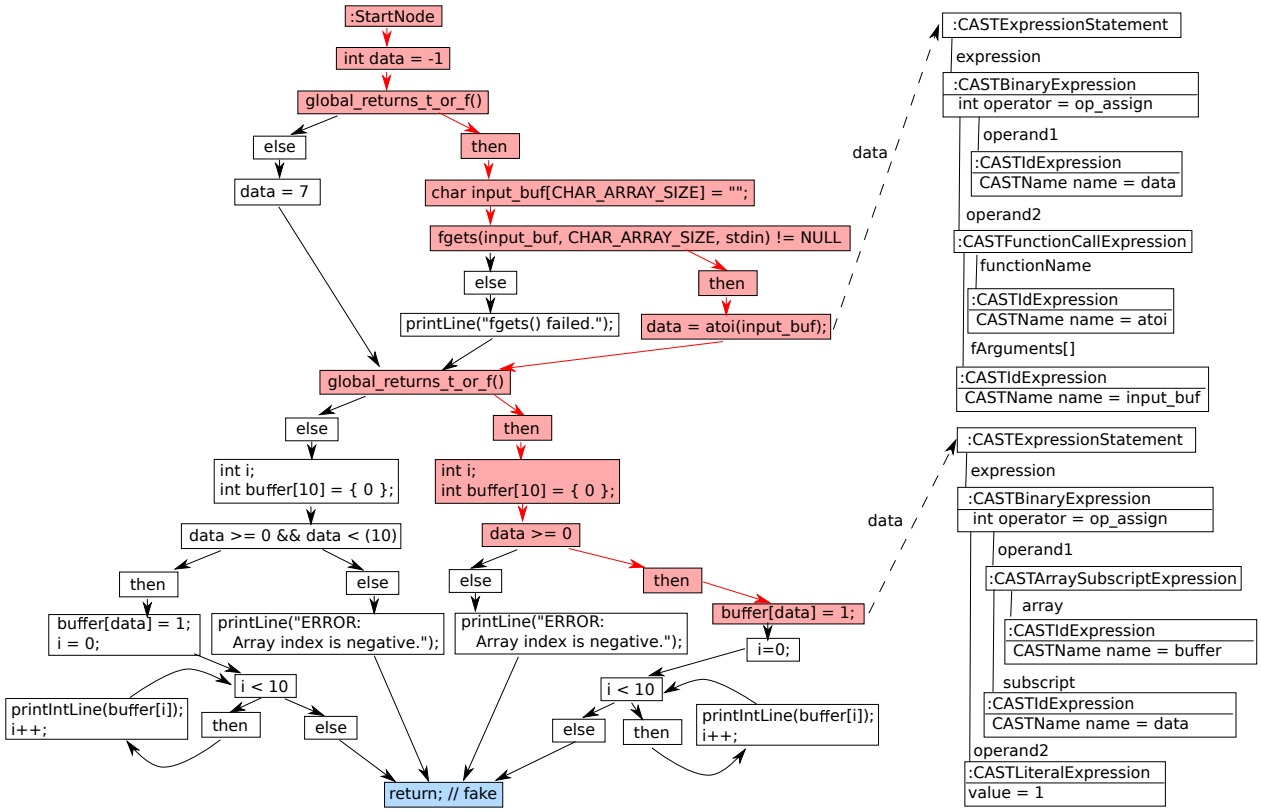


Figure 2. Left: Control flow graph for an example function from [14]. One path leading to a buffer overflow bug is marked red. Paths are merged at function exit (the function's exit node is marked blue). Right: Two AST subtrees referenced by CFG nodes

function's CFG start node, and the reduced local context is built from the variable declaration actions. The reduced local context is yielded by removing all writes to variables if the variables do not have global scope, are not written through pointers and are not the current function's return value. This approach does not necessitate a comparably more complex dead/live variable analysis. Path merge possibilities are detected using a class `ContextCache`, which is a `HashSet`. The keys are exit nodes with function call context, the values are the observed reduced local contexts. The context cache is queried at each function exit (CFG exit node). Comparing the reduced local contexts does not necessitate expensive calls to the SMT solver.

Path merging applies in the same way to branches that belong to loops, when the loop iteration number depends on program input (otherwise there would be only one satisfiable sub-path through the loop). Symbolic execution is currently applied with loop unrolling up to a maximum loop depth bound. A path through a loop can therefore split into a maximum number of paths equal to the loop unrolling bound. Branch nodes in the CFG belonging to loop statements are treated by symbolic execution just as branch nodes belonging to `if`/`else` statements. The branch nodes also have the same labels, i.e., '`then`' for the loop body and '`else`' to skip the loop. The only difference is that loops have a connector node with two incoming branches, which closes the loop before the decision node. However, this has no influence on the merging of unrolled paths.

V. Example Bug Checkers

A. Memory access

The major memory access bugs are stack-based or heap-based buffer over-write (overflow), over-read, under-write or under-read (CWE-121,122,124,126,127). The class `BoundsChecker` is triggered when the translation encounters array subscript expressions and pointer dereferences [12]. The bounds checker queries the set of equations, on which the pointer and offset variables depend, from the interpreter. Two satisfiability checks are then added to this equation system slice. One of them checks whether the offset could be negative (lower bound violation), the other checks whether the offset could be larger than the array size. These satisfiability queries are decided by the SMT solver.

B. Infinite loops

If an infinite loop can be triggered by unexpected program input, which is not properly validated, it can be used by an attacker for a denial of service attack. Since the standard number formats are discrete and finite, any infinite loop orbit without number overflow must be periodic. The common weakness enumeration calls this 'loop with unreachable exit condition' (CWE-835), in contrast to 'number overflow' bugs. An infinite loop can therefore be detected with a fixed-point satisfiability check. It checks whether it is satisfiable that the loop is re-visited with identical context. The class `InfiniteLoopChecker` is trigger for 'loop closed' events, i.e., when a decision node is re-visited on the path [16]. The

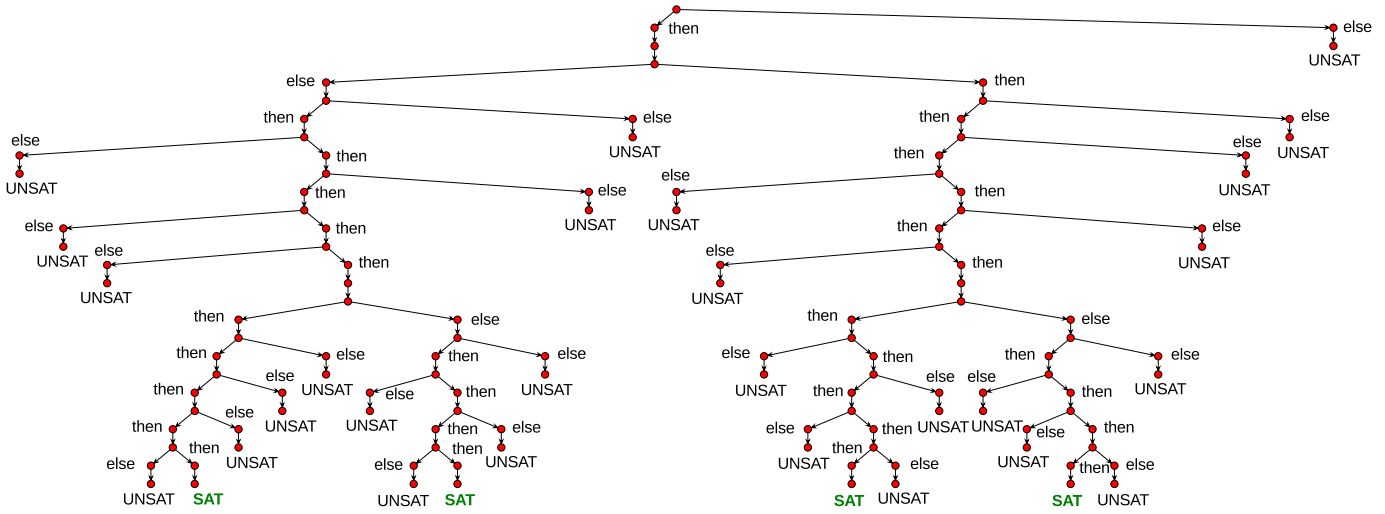


Figure 3. Execution tree for test program `CWE121_Stack_Based_Buffer_Overflow__char_type_overrun_memcpy_12` from [14], showing only decision and branch nodes.

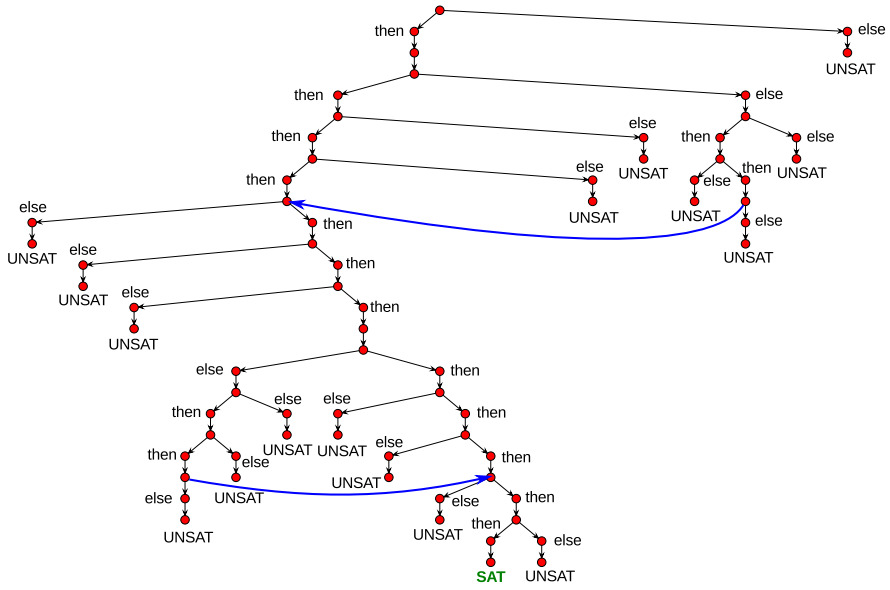


Figure 4. Effect of path merging for the test program of Figure 3. The execution tree is folded at two locations (blue arrows). The number of traversed satisfiable paths is reduced from four to one.

loop variables are identified using the `ActionLog`. The loop checker formulates the fixed-point query, which is then passed to the solver.

The loop checker avoids re-checking 'simple' loops in different contexts by performing context-free termination and non-termination checks at the loop's first closing event. This check is performed without the constraint of the path on which the loop is reached, i.e., only using the loop guard set and the unrolled loop body (unrolled one iteration). The termination check for 'simple' loops is based on Brouwer's fixed-point theorem [16], [58]. This theorem implies that all linear loops that do not have a fixed-point in the guard set must terminate.

Loops that have not been decided by the context-free checks are checked context-sensitively during symbolic execution, i.e., with consideration of the path and variable constraints of the path on which the loop is reached. Symbolic execution unrolls all satisfiable paths through the loop, up to a configurable loop depth bound. Like for other checkers with symbolic execution, bug detection is sound (no false positives) and bounded complete [16]. The infinite loop checker detects all infinite loops with t prefix loop iterations and a loop orbit periodicity of $p \leq n - t$, when all loops are unrolled up to a depth n [16].

C. Information flow

Examples for information flow bugs are cleartext transmission of sensitive information (CWE-319) or information exposures through environment variables, debug log files or shell error messages (CWE-526, 534, 535 [2]). The model for secure information flow follows the lattice model from [59]. Information belongs to 'security classes', and information must not flow from higher to lower security classes. The implemen-

tation regards the operating system API as trust boundary [17]. Information flows through a program from sources to sinks, which are standard library calls and therefore trust boundaries. These trust boundaries are implemented in the function models, which are accessed through the `Environment` classs (compare Figure 1). Program input is labelled with a security class. During symbolic interpretation, security class affiliation is inferred over the data flow. The `InfoFlowChecker` is triggered when a trust boundary is crossed with program output, i.e., for information sinks. The checker assures that sensitive output parameters do not flow into a lower security sink [17].

D. Atomicity violations

The detection is based on thread interleavings, that is, possible thread schedulings on a single-core processor. The detection is implemented on the abstraction level of thread interactions, from the recorded satisfiable program paths with 'lowest thread-id first' scheduling [18]. Alternative thread interleavings are generated by class `InterleavingGenerator`, and the detection of atomicity violations (CWE-366) is implemented in class `RaceChecker` (compare Figure 1). The `InterleavingGenerator` generates the scheduling tree of relevant alternative thread interleavings from the abstracted satisfiable program paths. The algorithm uses ample set partial order reduction [60] and selectable interleaving coverage [61].

1) Ample set partial order reduction: From a satisfiable program path, all other thread interleavings corresponding to different scheduling decisions can be generated. The generated set of interleavings should be of minimal size without degrading the ability to detect atomicity violations. The tree of possible scheduling decisions is traversed on-the-fly with depth-first search. The tree nodes are constructed as maximal sets of independant actions (ample sets). The construction of ample sets reduces the width of the scheduling tree and thus the number of generated interleavings. `read()` or `write()` actions from different threads for shared variables are independent if the variable is not the same. Actions may be blocked until they are enabled by other actions. Examples are a thread waiting to acquire a lock held by another thread, or a thread waiting to join another. Interleaving representatives are found as tree leafs, i.e., when there are no more blocked and enabled actions. The representative is given as path through the ample set scheduling tree [18].

2) Interleaving coverage: Like there are different code coverage criteria for single-threaded code, e.g., branch coverage or modified condition/decision coverage, there are also different interleaving coverage criteria for multi-threaded code [61]. In general, concurrency bugs can involve any number of threads and variables. However, due to its practical relevance, the special case of atomicity violations as overlapping `read()`/`write()` actions to the same variable from different threads is of particular interest. Therefore, the interleaving generation not only supports the 'all interleavings' criterion (with partial order reduced implementation), but currently also the 'local-or-remote-define' criterion from [61]. This criterion means that for every read access to a shared variable, both an interleaving where the respective variable was defined in the local thread and one interleaving where it was defined by a remote thread are covered. This criterion offers a far better scaling behaviour, at the expense of missing more involved concurrency bugs. Reduced interleaving coverage is implemented jointly with partial order reduction as 'branch and bound' pruning of the ample set scheduling tree.

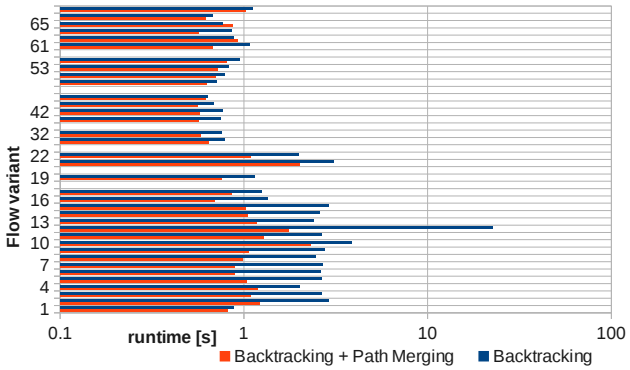
3) Atomicity violation detection: Atomicity violations are detected in the set of generated interleaving representatives by the class `RaceChecker`, which looks for overlapping `read()`/`write()` actions (at least one read and two writes) from different threads to the same variable [18].

VI. Experiments

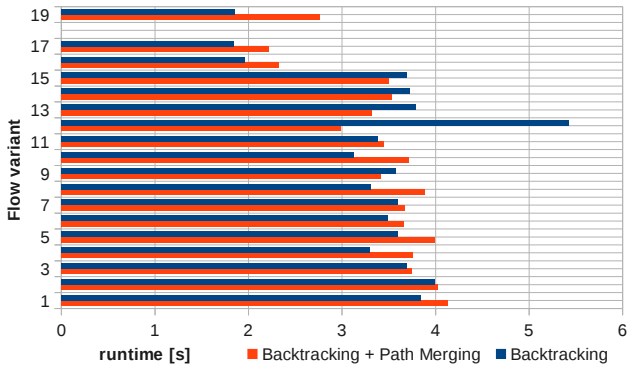
The tool is evaluated with test cases from the Juliet suite for C/C++ [14]. The test programs are artificial and automatically generated by combination of baseline bugs and control/data flow variants, in order to cover all language constructs. In the current version (1.2), the suite covers 1617 baseline bugs (flaw types) for 118 common weaknesses. Combined with 48 flow variants (38 of them for C, 10 only for C++) this results in over 60000 test cases (buggy programs) with together over 8 million lines of code. Bugs are context sensitive. The maximum bug context depth needed for accurate detection, i.e., no false positive and no false negative detections, is 5 function calls in 5 different source files (flow variant 54). Flow variants include flow controlled by global variables, different loop types, function pointers or void pointers,

Analysis of test programs could be started manually through the Eclipse GUI, i.e., through the extensions provided by Codan [52]. Codan in turn calls the symbolic execution plug-in presented in this paper (if activated in the Codan configuration). A screenshot for bug reporting with the CDT GUI is shown in Figure 6. In order to measure analysis run-times, the tests are rather run as JUnit plug-in tests. The measurements are obtained with Eclipse 4.3 on 64bit Linux kernel 3.2.0 and an i7-4770 CPU. This section evaluates the effect of path merging on analysis run-times. The same bug detection accuracy with and without path merging is validated, there are no false positive or false negative bug detections on the test set.

The effect of path merging on the execution tree is illustrated with the test program `CWE121_Stack_Based_Buffer_Overflow__char_type_overrun_memcpy_12`, which denotes a buffer overflow with `memcpy()` and flow variant 12 [14]. It contains a 'good' and a 'bad' function. The 'bad' function is shown in a slightly simplified version in listing 1. The function contains an `if`/`else` decision for which both branches are satisfiable. In the `then` branch it contains a buffer overflow bug, which is marked with a comment in the listing. For both branches the function only writes to stack variables, and the reduced local context at function exit is the empty set for both branches. Merging the two paths at function exit, which have split at the decision node, is therefore clearly possible without missing any bug. The 'good' function is almost identical, but is bug-free. Apart from some output functions, this program calls the 'good' and 'bad' function once each. Therefore, it contains four satisfiable paths. The execution tree is illustrated in Figure 3. The figure only shows decision nodes and branch nodes. Therefore, the top node in the figure is the first decision



(a) Analysis time for buffer overflow tests with `fgets()`.



(b) Analysis time for race condition tests on global variables.

Figure 5. Analysis run-times for test sets `CWE121_fgets` and `CWE366_global_int` from [14].

TABLE I. Analysis runtime sums for five test sets from [14], with and without path merging.

	CWE121_fgets (36 test programs)	**CWE121_memcpy** (18 test programs)	**CWE366_global_int** (18 test programs)	**CWE366_int_byref** (18 test programs)	**CWE835** (6 test programs)	**Sum** (96 test programs)
backtracking	80,7 s	14,7 s	61,2 s	62,1 s	9,0	227.7 s
backtracking + path merging	34,4 s	15,3 s	59.2 s	62.6 s	9,8	181.3 s

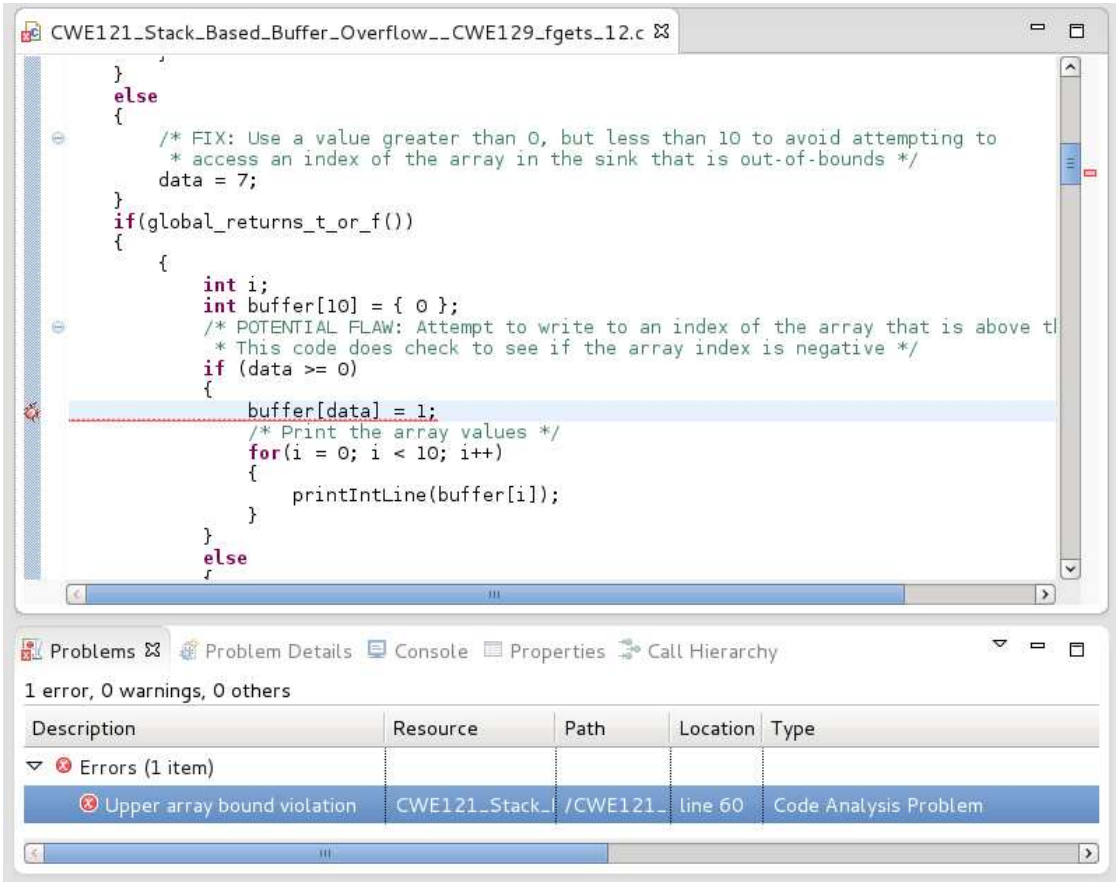


Figure 6. Bug reporting in the Eclipse GUI.

node after start of the `main()` function. Branch nodes are marked with the branch label (`if` or `else`). If a branch is decided to be unsatisfiable, the figure indicates this with a child note marked `UNSAT`. If the program end is reached, the figure indicates this with a child node marked `SAT`. Figure 4 shows the same tree when path merging is applied. Paths are merged at two points (the two function exits), which is indicated in the tree with blue arrows. An arrow connects two paths, which are merged. The arrow source indicates the subtree that is pruned, the arrowhead indicates the subtree that is further traversed. It can be seen that path merging corresponds to folding of the execution tree, the number of traversed satisfiable paths is reduced from four to one.

An infinite loop example from [14] is shown in Listing 2. The function contains a single-path loop, which is non-terminating for all input. While path merging could in principle be applied to infinite loops, a checker is still used to detect the infinite loop as software bug. The example loop is decided with the context-free non-termination test for 'simple loops' [16], unrolling the loop only once (rather than 256 times).

Listing 1. Simplified example function from [14], contains a buffer overflow in the `then` branch.

```
typedef struct _charvoid
{
    char x[16];
    void * y;
    void * z;
} charvoid;

void CWE121_memcpy_12_bad_simplified() {
  if(global_returns_t_or_f()) {
    charvoid cv_struct;
    cv_struct.y = (void *)SRC_STR;
    /* FLAW: Use the sizeof(cv_struct) which
       will overwrite the pointer y */
    memcpy(cv_struct.x, SRC_STR,
       sizeof(cv_struct));
    /* null terminate the string */
    cv_struct.x[(sizeof(cv_struct.x)/sizeof(
       char))-1] = '\0';
  }
  else {
    charvoid cv_struct;
    cv_struct.y = (void *)SRC_STR;
    /* FIX: Use sizeof(cv_struct.x) to avoid
       overwriting the pointer y */
    memcpy(cv_struct.x, SRC_STR,
       sizeof(cv_struct.x));
    /* null terminate the string */
    cv_struct.x[(sizeof(cv_struct.x)/sizeof(
       char))-1] = '\0';
  }
}
```

Table I shows analysis benchmark results with and without path merging for five test sets (in sum 96 programs) from [14], which contain buffer overflows, races and infinite loops bugs. The effect of path merging varies per test-set. Path merging requires a certain overhead for computing and comparing reduced local contexts. If path merging possibilities are found, there is a speed-up of analysis time. The table shows a 60% speed-up for tests containing buffer overflows with `fgets()`

Listing 2. Example infinite loop from [14].

```
void CWE835_Infinite_Loop__do_01_bad() {
  int i = 0;
  do {       /* FLAW: no break */
    printIntLine(i);
    i = (i + 1) % 256;
  } while(i >= 0);
}
```

(from 80.7 s to 34.4 s), but a 9% slow-down for the infinite loop tests.

Figure 5 shows the analysis runtimes for the sets of buffer overflows with `fgets()` (Figure 5a) and for the races test set on global variables (Figure 5b). The figure shows the runtimes depending on the test case data/control flow variant, for the symbolic execution engine with backtracking only and for backtracking with path merging. Figure 5a uses a logarithmic scale and contains values for 36 flow variants. Flow variants in Juliet are not numbered consecutively, to leave room for later insertions. Since path merging folds complete subtrees of a program's execution tree, it has an exponential effect on runtimes. This is exemplified by flow variant 12. While merging paths for the `memcpy()` buffer overflow with variant 12 reduces the runtime only from 1.1 s to 0.8 s, the runtime for the `fgets()` buffer overflow is reduced from 22.8 s (longest analysis time for any tested program) to 1.7 s. This is because the `fgets` test program contains several other decision nodes with two satisfiable branches.

The dependence of possible path merging speed-up on the program structure becomes clear through the specific test case structure, which is a combination of baseline flaws with flow variants. There are three possibilites:

1) The baseline flaw has a path merging possibility. Then it is likely that there is a speed-up already for the simplest test program containing the bug (flow variant 1), and for all other flow variants. An example is the `CWE121_fgets` test set (compare Figure 5a).
2) The baseline flaw does not have a path merging possibility. Then there can only be a speed-up for test cases, in which the flow variant contains a merging possibility. In the current test suite version this is only the case for flow variant 12. An example is the `CWE366` test set, where only flow 12 shows a significant path merging speed-up (compare Figure 5b).
3) Neither the baseline flaw nor any flow variants compatible with this flaw contain merging possibilites. An example is the infinite loop test set (`CWE835`, compare Table I).

VII. Discussion

This paper describes a backtracking symbolic execution engine with path merging functionality and its implementation in Eclipse CDT. Symbolic execution enables sound bug detection, i.e., without false positives. The evaluation of path merging with small test programs from the Juliet suite already shows a significant speedup. For larger programs, path merging has an exponential effect on analysis runtimes (exponential in

the number of decision nodes with more than one satisfiable branch). Future work might include the investigation of the effect of additional merge points inside functions. Automated abstraction might enable scaling to larger programs by offering yet more path merging possibilities. The implementation could also be used as a basis for selective symbolic execution, e.g., by adding consistent concrete execution using CDT's debugger services framework. Another direction is the automated generation of quick-fix refactoring proposals based on the obtained information about bugs and program paths on which they occur. The tight tool integration enabled by Eclipse seems advantageous for this purpose.

ACKNOWLEDGEMENT

This work was funded by the German Ministry for Education and Research (BMBF) under grant 01IS13020.

REFERENCES

[1] A. Ibing, "A backtracking symbolic execution engine with sound path merging," in Int. Conf. Emerging Security Information, Systems and Technologies, 2014, pp. 180–185.

[2] R. Martin, S. Barnum, and S. Christey, "Being explicit about security weaknesses," CrossTalk The Journal of Defense Software Engineering, vol. 20, 3 2007, pp. 4–8.

[3] J. King, "Symbolic execution and program testing," Communications of the ACM, vol. 19, no. 7, 1976, pp. 385–394.

[4] L. deMoura and N. Bjorner, "Satisfiability modulo theories: Introduction and applications," Communications of the ACM, vol. 54, no. 9, 2011, pp. 69–77.

[5] C. Barrett, A. Stump, and C. Tinelli, "The SMT-LIB standard version 2.0," in Int. Workshop Satisfiability Modulo Theories, 2010.

[6] L. deMoura and N. Bjorner, "Z3: An efficient SMT solver," in Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2008, pp. 337–340.

[7] C. Cadar, D. Dunbar, and D. Engler, "KLEE: Unassisted and automatic generation of high-coverage tests for complex systems programs," in USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2008, pp. 209–224.

[8] C. Lattner and V. Adve, "LLVM: A compilation framework for lifelong program analysis and transformation," in Int. Symp. Code Generation and Optimization (CGO), 2004, p. 75.

[9] T. Kremenek, "Finding software bugs with the Clang static analyzer," LLVM Developers' Meeting, Aug. 2008, retrieved: 05/2015. [Online]. Available: http://llvm.org/devmtg/2008-08/Kremenek_StaticAnalyzer.pdf

[10] F. Nielson, H. Nielson, and C. Hankin, Principles of Program Analysis. Springer, 2010.

[11] C. Cadar, V. Ganesh, P. Pawlowski, D. Dill, and D. Engler, "EXE: Automatically generating inputs of death," in 13th ACM Conference on Computer and Communications Security (CCS), 2006, pp. 322–335.

[12] A. Ibing, "Parallel SMT-constrained symbolic execution for Eclipse CDT/Codan," in Int. Conf. Testing Software and Systems (ICTSS), 2013, pp. 196–206.

[13] P. Boonstoppel, C. Cadar, and D. Engler, "RWset: Attacking path explosion in constraint-based test generation," in Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2008, pp. 351–366.

[14] T. Boland and P. Black, "Juliet 1.1 C/C++ and Java test suite," IEEE Computer, vol. 45, no. 10, 2012, pp. 88–90.

[15] TIOBE index, retrieved: 05/2015. [Online]. Available: www.tiobe.com/index.php/content/paperinfo/tcpi

[16] A. Ibing, "A fixed-point algorithm for automated static detection of infinite loops," in IEEE Int. Symp. High Assurance Systems Eng., 2015, pp. 44–51.

[17] P. Muntean, C. Eckert, and A. Ibing, "Context-sensitive detection of information exposure bugs with symbolic execution," in Int. Workshop Innovative Software Development Methodologies and Practices, 2014, pp. 84–93.

[18] A. Ibing, "Path-sensitive race detection with partial order reduced symbolic execution," in Workshop on Formal Methods in the Development of Software, 2014, pp. 311–322.

[19] C. Cadar and K. Sen, "Symbolic execution for software testing: Three decades later," Communications of the ACM, vol. 56, no. 2, 2013, pp. 82–90.

[20] C. Cadar et al., "Symbolic execution for software testing in practice – preliminary assessment," in Int. Conf. Software Eng., 2011, pp. 1066–1071.

[21] C. Pasareanu and W. Visser, "A survey of new trends in symbolic execution for software testing and analysis," Int. J. Software Tools Technology Transfer, vol. 11, 2009, pp. 339–353.

[22] S. Anand, E. Burke, T. Chen, J. Clark, M. Cohen, W. Grieskamp, M. Harman, M. Harrold, and P. McMinn, "An orchestrated survey of methodologies for automated software test case generation," Journal of Systems and Software, vol. 86, no. 8, 2013, pp. 1978–2001.

[23] P. Boudin, P. Cuoq, J. Filliatre, C. Marche, B. Monate, Y. Moy, and V. Prevosto, "ACSL: ANSI/ISO C specification language, version 1.9," 2013, retrieved: 05/2015. [Online]. Available: http://frama-c.com/download/acsl.pdf

[24] L. Burdy, Y. Cheon, D. Cok, M. Ernst, J. Kiniry, G. Leavens, K. Leino, and E. Poll, "An overview of JML tools and applications," International Journal on Software Tools for Technology Transfer, vol. 7, no. 3, 2005, pp. 212–232.

[25] L. Correnson, P. Cuoq, F. Kirchner, V. Prevosto, A. Puccetti, J. Signoles, and B. Yakubowski, "Frama-C user manual, release sodium," 2015, retrieved: 05/2015. [Online]. Available: http://frama-c.com/download/frama-c-user-manual.pdf

[26] E. Cohen, M. Dahlweid, M. Hillebrandt, D. Leinenbach, M. Moskal, T. Santen, W. Schulte, and S. Tobies, "VCC: A practical system for verifying concurrent C," in Int. Conf. Theorem Proving in Higher Order Logics, 2009, pp. 23–42.

[27] E. Clarke, O. Grumberg, and D. Peled, Model Checking. MIT Press, 1999.

[28] E. Clarke, D. Kroening, and F. Lerda, "A tool for checking ANSI-C programs," in Tools and Algorithms for the Construction and Analysis of Systems, 2004, pp. 168–176.

[29] D. Detlefs, K. Leino, G. Nelson, and J. Saxe, "Extended static checking," SRC Research report 159, Compaq Systems Research Center, 1998.

[30] P. Cousot and R. Cousot, "Abstract interpretation: A unified latttice model for static analysis of programs by construction or approximation of fixed points," in Symp. Principles of Programming Languages (POPL), 1977, pp. 238–252.

[31] K. Zurowska and J. Dingel, "Symbolic execution of UML-RT state machines," in ACM Symp. Applied Computing, 2012, pp. 1292–1299.

[32] H. Hansen, J. Ketema, B. Luttik, M. Mousavi, and J. Pol, "Towards model checking executable UML specifications in mCRL2," Innovations Syst. Softw. Eng., no. 6, 2010, pp. 83–90.

[33] J. Abrial and L. Mussat, "Introducing dynamic constraints in B," in B Conference, 1998, pp. 83–128.

[34] E. Reisner, C. Song, K. Ma, J. Foster, and A. Porter, "Using symbolic evaluation to understand behaviour in configurable software systems," in Int. Conf. Software Eng., 2010, pp. 445–454.

[35] N. Tillmann and J. Halleux, "Pex – white box test generation for .NET," in Int. Conf. Tests and Proofs (TAP), 2008, pp. 134–153.

[36] W. Visser, C. Pasareanu, and S. Khurshid, "Test input generation with Java PathFinder," in Int. Symp. Software Testing and Analysis (ISSTA), 2004, pp. 97–107.

[37] D. Song, D. Brumley, H. Yin, J. Caballero, I. Jager, M. Kang, Z. Liang, J. Newsome, P. Poosankam, and P. Saxena, "BitBlaze: A new approach to computer security via binary analysis," in Int. Conf. Information Systems Security, 2008, pp. 1–25.

[38] D. Brumley, I. Jager, T. Avgerinos, and E. Schwartz, "BAP: A binary

analysis platform," in Int. Conf. Computer Aided Verification, 2011, pp. 463–469.

[39] P. Godefroid, M. Levin, and D. Molnar, "Automated whitebox fuzz testing," in Network and Distributed System Security Symp. (NDSS), 2008.

[40] K. Sen, D. Marinov, and G. Agha, "CUTE: A concolic unit testing engine for C," in European Software Engineering Conference and International Symposium on Foundations of Software Engineering, 2005, pp. 263–272.

[41] P. Godefroid, N. Klarlund, and K. Sen, "DART: Directed automated random testing," in Conference on Programming Language Design and Implementation, 2005, pp. 213–223.

[42] V. Chipounov, V. Kuznetsov, and G. Candea, "S2E: A platform for in-vivo multi-path analysis of software systems," in Int. Conf. Architectural Support for Programming Languages and Operating Systems, 2011.

[43] V. Kuznetsov, J. Kinder, S. Bucur, and G. Candea, "Efficient state merging in symbolic execution," in Conf. Programming Language Design and Implementation (PLDI), 2012, pp. 193–204.

[44] S. Graf and H. Saidi, "Construction of abstract state graphs with PVS," in Int. Conf. Computer Aided Verification (CAV), 1997, pp. 72–83.

[45] E. Clarke, O. Grumberg, Y. Lu, S. Jha, and H. Veith, "Counterexample-guided abstraction refinement for symbolic model checking," Journal of the ACM, vol. 50, no. 5, 2003, pp. 752–794.

[46] J. Jaffar, A. Santosa, and R. Voicu, "An interpolation method for CLP traversal," in Int. Conf. Principles and Practice of Constraint Programming (CP), 2009, pp. 454–469.

[47] K. McMillan, "Lazy annotation for program testing and verification," in Int. Conf. Computer Aided Verification (CAV), 2010, pp. 104–118.

[48] J. Jaffar, J. Navas, and A. Santosa, "Unbounded symbolic execution for program verification," in Int. Conf. Runtime Verification, 2011, pp. 396–411.

[49] J. Jaffar, V. Murali, J. Navas, and A. Santosa, "TRACER: A symbolic execution tool for verification," in Int. Conf. Computer Aided Verification (CAV), 2012, pp. 758–766.

[50] W. Craig, "Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory," The Journal of Symbolic Logic, vol. 22, no. 3, 1957, pp. 269–285.

[51] OSGi Alliance Specifications, retrieved: 05/2015. [Online]. Available: www.osgi.org/Specifications

[52] A. Laskavaia, "Codan- C/C++ static analysis framework for CDT," in EclipseCon, 2011.

[53] D. LeBerre and A. Parrain, "The SAT4J library, release 2.2, system description," Journal on Satisfiability, Boolean Modeling and Computation, no. 7, 2010, pp. 59–64.

[54] R. Dannenberg and G. Ernst, "Formal program verification using symbolic execution," IEEE Trans. Software Eng., vol. 8, no. 1, 1982, pp. 43–52.

[55] T. Parr, Language Implementation Patterns. Pragmatic Bookshelf, 2010.

[56] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.

[57] E. Clarke, O. Grumberg, M. Minea, and D. Peled, "State space reduction using partial order techniques," Int. J. Software Tools for Technology Transfer, vol. 2, no. 3, 1999, pp. 279–287.

[58] L. Brouwer, "Über Abbildungen von Mannigfaltigkeiten," Mathematische Annalen, no. 71, 1911.

[59] D. Denning, "A lattice model of secure information flow," Communications of the ACM, vol. 19, no. 5, 1976, pp. 236–243.

[60] D. Peled, "Combining partial order reduction with on-the-fly model-checking," in Workshop on Computer Aided Verification, 1994.

[61] S. Lu, W. Jiang, and Y. Zhou, "A study of interleaving coverage criteria," in ECEC/FSE, 2007.

Reflections on Evolving Large-Scale Security Architectures

Geir M. Køien

Institute of ICT
Faculty of Engineering and Science
University of Agder, Norway
Email: geir.koien@uia.no

***Abstract*—In this paper, we conduct an informal analysis of evolving large-scale security architectures. The 3rd generation partner project (3GPP) mobile systems is our example case and we shall investigate how these systems have evolved and how the security architecture has evolved with the system(s). The 3GPP systems not only represent a truly long-lived system family, but are also a massively successful system family. What once was an auxiliary voice-based infrastructure has evolved to become a main, and thereby critical, information and communications technology (ICT) infrastructure for billions of people. The 25+ years of system evolution has not all been a linearly planned progression and the overall system is clearly also a product of its history. The goal of this paper is to capture some of the essence of security architecture evolution for critical ICT systems. What makes the evolution work and what may break it? These are important issues to analyse, and this paper aim at highlighting some of the aspects that play a role in security architecture evolution. In this sense, the paper is about research directions.**

Keywords–Evolving Security; System Security; Security Architecture; Long-term security planning; Migration; Mitigation; Deprecation.

I. INTRODUCTION

This paper is an extended version of the paper "Challenges for Evolving Large-Scale Security Architectures" [1] (SecurWare 2014). The scope has been broadened and significant extensions has been made. In particular, we have added new material to Sections III and V, and Section VI is entirely new. Other amendments have been made throughout the paper.

A. Background and Motivation

The example system investigated in our study, the 3GPP systems, has gradually become important, all-encompassing and pervasive on a global scale. Initially, the systems only served as an auxiliary and adjunct infrastructure, but gradually it has replaced the fixed line telephone. Today, the 3GPP mobile system services are pervasive and ubiquitous, and the systems have also become a major IP infrastructure. The convenience of mobility has been a major driver, and now the mobile infrastructures are poised to become the major access network for the fast growing Internet-of-Things (IoT) and the machine-to-machine (m2m) ecosystems.

Security was never a big priority in the 1G analog cellular systems. Originally, there were not any security problems either, but with opportunity and almost non-existent protection came theft and fraud. So, when the 2G digital systems came, the need for security was recognized and one devised mechanisms to address the threats from the first generation [2].

When the 3G architecture was designed, there were no serious practical problems with 2G security as such. However, it was clear that the authentication was too weak, that 64-bit encryption was not going to be enough and that the scope would not suffice for IP connectivity [3]. The 3G security model therefore improved on existing schemes to address the known shortcomings [4]. Additionally, one added support for core network protection (profiled use of IPsec) [5]. With the advent of 4G security one found that weaknesses induced by backwards compatibility with 2G was a most urgent problem to be fixed. Of course, one also took the opportunity to fix some of the other shortcomings of 3G security. To this end, the key-deriving key hierarchy in 4G was a clear improvement, both security-wise and performance-wise [6]. During the progression from 3G to 4G a lot of core network protection measures were added, but these were not so much part of a design as a patchwork of useful, but specific schemes. In the meantime, the importance of the systems have far outgrown the added protection capabilities, and as technology and scope has progressed it is clear that the 3GPP security architecture has not really evolved far enough to cater for the new needs.

The 3GPP security has rightly been criticised in academic circles. However, much of the criticism is somewhat misguided. That is, the criticism may technically be accurate. However, the impact of a theoretical cryptological weakness are often negligible in practice, and the suggested improvements are often utterly impractical to deploy in an existing system. To neglect to cater for migration is a major showstopper in practical terms. Obviously, there is also organizational politics at play, which makes it even harder to make changes, particularly if the benefits are perceived to be minor. The proverb "*If it ain't broke, don't fix it*" comes to mind.

To criticize the 3GPP security architecture is one thing, but it is rather more interesting to try to investigate the ways forward. In particular, how does a security architecture evolution actually work. What can we learn from the 3GPP case and how can we use this to make it work better. The goal is not completeness or a full understanding, but rather to identify key aspects that define evolving security architectures.

Security is difficult and hard to get right. Good cryptographic primitives are very hard to design and it is even harder to verify that they do not have any fatal flaws. Still, by and large, this is doable. Cryptographic protocols are also very hard to get right. We know quite a lot about how to construct communication protocols, and there are many formal verification methods that allow us check for a whole range of properties. Still, it is very hard to design a secure cryptographic

protocol. There are tools that will allow us to check security protocols (see [7, 8]) for certain security properties, but overall the state of the art for security protocol design is immature [9].

When it comes to security architectures, we are often dealing with complex systems that needs to be secured. The problem is highly complex and it must be dealt with on many different levels. This problem is not well understood, and yet it is vital that these complex system will be properly protected. This situation the motivation for this paper. It must be seen as an initial effort. We hope to improve on situation awareness and we hope to inspire more future work in how security architectures evolve. This, we believe, will provide us with tools to make better, more resilient and robust systems.

B. The 3GPP System Context

Mobile radio existed before we got fully automated systems. Systems like the analog 1G Nordic Mobile Telephony system, which had unassisted call setup and automatic handovers, marks the start of true cellular systems around 1980. The first 3GPP system is the second generation (2G) Global System for Mobile communications (GSM), developed in the mid/late 1980ies. Originally, GSM only featured circuit-switched (CS) services, but was later adapted to also include packet-switched (PS) services through the General Packet Radio Service (GPRS) extension. With the new millennium came the third generation (3G) Universal Mobile Telecommunications System (UMTS), which natively features both CS and PS services. From around 2010 we also have the fourth generation (4G) Long-Term Evolution (LTE) system, which is a broadband PS-only system. LTE is further developed into LTE-Advanced (LTE-A).

1) Principal Parties: From a subscriber perspective, the system can be described with three types of principal parties.

- The Home Public Land Mobile Network (HPLMN).
- The Visited Public Land Mobile Network (VPLMN).
- The subscriber/user (USER).

These parties also represent legal entities, and the relationships are determined by contractual agreements. It is immediately clear that while the number of HPLMN and VPLMN operators will be limited to a few thousand, the number of subscribers will easily be in the billions. There is also a distinction between a subscription and a legal entity in that a person or organization may own many subscriptions, and this will certainly be the case for IoT/m2m subscriptions.

A national telecom regulator will also be involved, in addition to external service providers. Over-national regulatory bodies also exists, but their influence will likely be mediated by the national regulator. One may also add intruders to the list. The external service providers usually have little influence on how the networks operate and so we exclude those for further discussion. Likewise, in this context, we do not see a need for including virtual mobile network operators (VMNOs).

2) System Development: The 3GPP system specifications are developed by the 3GPP, but ratification is done by the organizational partners (formal standardization bodies). The design is "*contribution-driven design-by-committee*", and the process is largely consensus driven. The contribution-driven aspect quite literally means that company impact is relative to the number of contributions. Normally, it will be enough if 4 companies sign up for commitment to develop a feature. By-and-large, there is no real way to stop initiatives, and so the architecture sometimes suffer from new developments that do not really fit well with the overall architecture. Initiatives to develop new features may of course be stopped, but this is more likely to be caused by patent issues etc. than related to system architectural concerns.

The impact is noticeable when it comes to priorities and efforts spent. Early on, when GSM/GPRS was specified, the operators took considerable responsibility and led many of the efforts. Subsequently, the vendors have taken over more and more of this work. The impetus to carry out work is clearly related to the business potential the work has. Unfortunately, investments in security functions seldom look like a good business proposition prior to an incident.

3) Mandatory Features: The 3GPP differentiates between *mandatory for implementation* and *mandatory for use*. That is, a feature may be mandatory to be implemented by the vendors if they want compliance with a system release. At the same time, the operators may freely disregard the feature if they want. Other functions may be mandatory both to develop and deploy. In terms of deployment, this often means that the features that are not mandatory for deployment will only get deployed at a later stage, if at all.

4) Scope: The 3GPP scope has extended over the years and so has the scope of the security protection. However, aspects such as server hardening and similar is still considered well outside the scope, and generally one limits the scope to the protocols directly developed by 3GPP or for features that are otherwise captured by 3GPP specifications. Except for where interoperability is at stake, one generally avoids schemes being *mandatory for use*.

5) Licenses and Regulatory Requirements: Cellular systems operate in licensed bands and are subject to regulatory requirements. These requirements include support for lawful interception (LI) and emergency call (EC) [10, 11]. The last decade we have also had anti-terrorist measures such the EU Data Retention Directive (DRD) [12].

C. Brief Introduction to 3GPP Systems

1) 2G – GSM and GPRS: The GSM and GPRS systems are the 2G systems. It is common to see monikers like 2.5G used for GPRS, and 2.9G used for GPRS with Enhanced Data rates for GSM Evolution (EDGE). The main GSM features are mobility, speech and text messaging. GPRS is an overlay system to GSM. It features two additional core network nodes and provides PS support. With EDGE (new codecs) it provides up to 236 kbps data-rate. There is also an "Evolved EDGE" extension on the horizon, with yet higher data-rates. The 2G-based radio access network is called GSM EDGE Radio Access Network (GERAN).

2) 3G – UMTS (incl. High-Speed Packet Access (HSPA)): The UMTS system was finalized in late 1999 and is a combined CS/PS system. It can readily achieve $>$10 Mbps data-rates (w/max. rates $>$100 Mbps downlink). The system is a mix

of GSM/GPRS technology and protocols and, increasingly, IP-based protocols and technology. The radio access network is called the Universal Terrestrial Radio Access Network (UTRAN).

3) 4G – LTE and LTE-A: The LTE systems are designed as all-IP networks (AIPN) and features true mobile broadband. The core network is fully IP based and there are no CS components in LTE. The radio system is highly advanced and provides true broadband services. The radio base-stations, called eNB, are logically mesh connected. There are no longer any controllers in the access network (E-UTRAN). The VPLMN mobility functions are carried out by the mobility management entity (MME) server.

D. Paper Layout

In Section II, we briefly outline the security of the 3GPP systems. In Section III, with investigating what evolution means in the context of a security architecture. Then we proceed in Section IV, were we discuss what may induce changes in a security architecture. This is followed up in Section V with some assumptions regarding the security architecture and the system context. In Section VI, we take a look at some of the factors that come into play and that might cause problems and even outright failure for a security architecture evolution. These factors are almost exclusively non-technical ones. In Section VII, we try to learn from the lessons and provide some advice. In Section VIII, we try to distil actionable knowledge from the previous sections. Finally, we sum up our effort and provide some concluding remarks in Section IX.

II. Security in the 3GPP Systems

In this section, we provide a short description of the main features of the 3GPP security provisions.

A. 2G Security

There is no well-defined security architecture per se in the 2G systems. The main security specification was technical specification (TS) 03.20 "Security-related network functions", which subsequently has been transposed into TS 43.020 [2]. It defines the identity- and location privacy scheme, the entity authentication protocol and the smart-card based security functions. It also outlines the over-the-air cipher function(s). The over-the-air ciphers must be supported both by all access networks and user equipment (UE). These ciphers must therefore be fully standardized. Figure 1 outlines the GSM security procedures. The scenario consists of the user equipment, the visited network and the home network.

1) Background and Requirements: In the voice-only 1G systems one had experienced charging fraud and impersonation fraud. Two distinct types of attacks quickly came into focus:

- **a)** Eavesdropping was a problem as the analogue voice channel was unprotected and easy to listen-in on.
- **b)** Faking the call setup signaling, which was digital, was quite easy and could in principle be done by simply recording a setup sequence and then later replay it.

A main priority for the second generation system GSM was therefore to **a)** protect the over-the-air channel against eavesdropping, such that it would no longer be the weakest link, and **b)** provide credible subscriber authentication to avoid impersonation attacks. The fact that GSM featured digitally encoded speech made protection much easier, as it permitted use of encryption.

2) The 2G Security Architecture: GSM security is based on a physical subscriber identity module (SIM). For portability reasons it was decided to use a smart-card. The SIM comprises both hardware and software functionality, and it contains the authentication and key agreement (AKA) functions (symmetric crypto). The SIM also contains the security credentials, like the permanent subscriber identity, the International Mobile Subscriber Identity (IMSI), and the corresponding 128-bit authentication secret, called K_I in the 2G SIM.

The AKA protocol used is called GSM AKA, and it is a single-pass challenge-response protocol with a signed response (SRES). The challenge is a pseudo-random 128-bit RAND bit-field and the response is the 32-bit SRES element. The challenge-response part is dependent on an "authentication set" forwarding stage, in which the HPLMN forwards the authentication credentials to the VPLMN network. The protocol runs between the SIM and the visited network. This scheme is efficient and allows for fast and simple authentication of the subscriber as well as deriving a session key (the 64-bit K_C). The SIM features the A3 and A8 AKA interfaces, which are only found in the SIM and the home subscriber database (HLR). The original example implementation of A3 and A8, called COMP128, is cryptographically broken [13], but still seems to be in use in many markets.

Over-the-air encryption is by means of the A5 stream cipher family, which is located in the mobile phone and the base transceiver station (BTS). There are several A5 versions available, but the original A5/1 is still the default and mandatory-to-deploy algorithm. It can easily be broken today by a dedicated attacker [14]. The breaking of A5/1 is based on a clever variant of applied brute-force and space/time trade-offs called a rainbow table attack. First, one essentially brute-force breaks A5/1 and stores the results in large tables. This is a once-only effort. The process is computationally very costly and also very time consuming, but modern graphics cards makes this feasible and even quite affordable. The process also requires considerable storage (terabytes), but this has become a commodity. Subsequently, one uses the stored tables and clever algorithms to derive the session keys. This second step is fast and computationally inexpensive.

The A5/2 algorithm, which was explicitly designed to be weak (CoCom regulations), is officially deprecated. The A5/3 algorithm, which is based on the 3G KASUMI design, is the current best option for GSM, but rainbow table attacks still work since the algorithm is limited to 64-bit [15]. The A5 family is based around a 64-bit key, expect the recent A5/4 cipher, which is a 128-bit design based on the KASUMI algorithm. In GPRS, one uses the GSM AKA protocol as-is, but here one uses the GPRS Encryption Algorithm (GEA) ciphers to protect the asynchronous packet transfers.

3) Omissions and Shortcomings: There are many obvious omissions and shortcomings to GSM security. This is not strange as the 2G systems do not have a security architecture as such; it is more akin to a collections or measures put together without well-defined requirements. The following list (derived

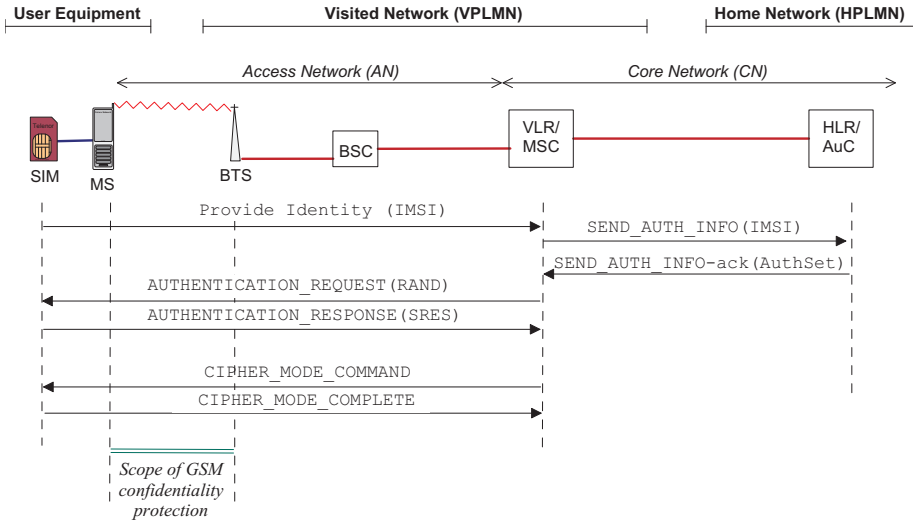


Figure 1: 2G Security: GSM security overview.

in [16]) identifies some of the flaws. Even with all these flaws, the GSM/GPRS system has been a remarkably secure system. However, some 25 years down the line and the shortcomings have become serious liabilities. There are also a number of implementations issues [17]. The list is not fair with regard to the threats found early on, but it is certainly valid now.

- One-way authentication is utterly inadequate.
- Delegated authentication is naive trust-wise.
- SIM/AuC: pre-shared authentication secrets is a liability.
- No inter-operator authentication.
- No way to authenticate system nodes.
- No uniqueness/freshness to challenges.
- Unauthenticated plain-text transfer of security credentials.
- Unprotected key transfer.
- Missing key binding and too short keys.
- Key refresh dependent of re-authentication.
- Missing expiry condition on security context.
- Weak A3/A8 functions and no key-deriving key structure.
- Short A5 key stream cycle and key stream re-use.
- Redundant and structured input to A5 (expand-then-encrypt).
- Highly redundant input to A5 (in signaling message).
- Protection coverage/range too short (only MS – BTS).
- Missing integrity protection.
- Weak/inadequate identity/location privacy.
- No core network control plane (signaling) security features.
- No core network user plane protection.
- No IP layer protection (GPRS).
- No mobile phone (MS) platform security.

B. 3G Security

1) Background and Requirements: Security in the UMTS system is described briefly in [16, 18] and in considerable depth in [19]. The main security specification is TS 33.102 [20]. A "Security Objectives and Principles" [4] background document was also provided, together with a threats and requirements analysis document [3]. One also introduced Network Domain Security (NDS), which includes IPsec profiles for use with 3GPP systems [5] and a standard set of public-key infrastructure (PKI) protocols and methods [21].

2) The 3G Security Architecture: The UMTS security architecture, depicted in Figure 2, is an important overhaul of the GSM security, yet the underlying system model remains much the same. Amongst the features are:

- New subscriber card (UICC) with security module (USIM).
- Introduction of 128-bit crypto primitives.
- Improved two-way'ish AKA algorithm (UMTS AKA).
- Introduction of core network protection (IP protocols).

Sadly, backwards compatibility concerns also dictated that the GSM SIM could still be used, which when used re-introduces many if not most of the 2G weaknesses.

3) The IP Multimedia Subsystem (IMS): IMS came with UMTS (Rel.5). We do not include IMS in our discussions as it is an optional service-level feature.

We note that a cut-down version of IMS will be used to support voice over LTE (VoLTE), and this version (IMS MMTel) will be important in 4G systems.

4) Omissions and Shortcomings: The 3G security is substantially better and more future proof than the 2G security, and one really has a security architecture. The architecture is by no means perfect or complete, but it does at least capture the main risks/threats and defines what one wants to protect. Completeness will always be an issue, but in the 3G systems we also have that there sometimes is a considerable mismatch between stated goal and what the mechanisms achieve. A case

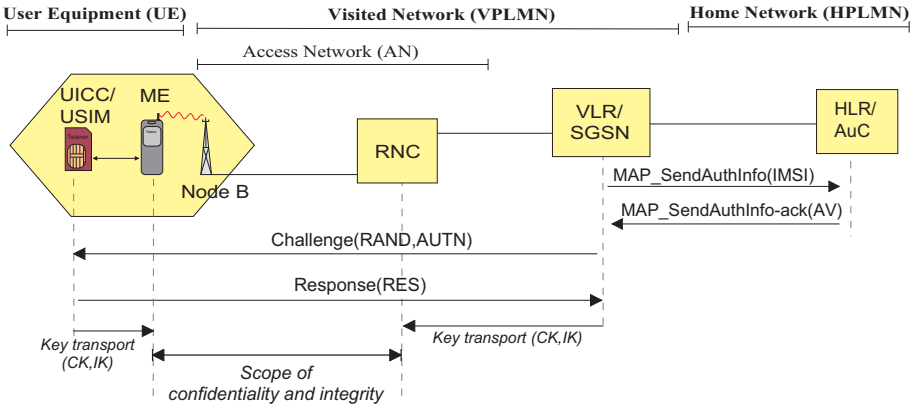


Figure 2: 3G Security: UMTS security architecture.

in point would be the identity/location privacy requirements, which does capture the problem well, but the mechanisms that should provide the necessary services are woefully inadequate. However, they are *a)* exactly the same as for the 2G systems and *b)* intimately tied to the identity presentation scheme defined in the basic mobility management (MM) protocol machinery (discussed in [16, 22]). The identity presentation scheme is weak security-wise on several levels, and there are also effective and efficient denial-of-service attacks against it [23, 24]. These problems cannot be remedied by tinkering and quick fixes as they are inherent to the system access procedures. Making changes to the access procedures would have been a major undertaking, and since there was considerable time pressure to complete the 3G standard, improvements to identity/location privacy simply did not happen (there were efforts investigating the possibilities during the Rel.99 design).

Many of the items on the 2G list of omissions and shortcomings are mitigated and resolved, but suffice to say that many of the 2G weaknesses were inherited or permitted through backwards compatibility requirements. Another main problem with 3G security is the limited scope.

C. 4G Security

1) Background and Requirements: The book "LTE Security" [25] is a good and thorough introduction to the topic. The main security standard for LTE is TS 33.401 [6]. LTE and LTE-A are very similar with respect to the security architecture, which for historical reasons is called the "System Architecture Evolution (SAE)" security architecture. The term Evolved Packet System (EPS) is also used.

The radio access architecture changed significantly with LTE, and this triggered large-scale changes to the whole system, including the security architecture. This, together with wholesale abandonment of non-IP based system protocols, marks a clear cut from previous practices. Despite these changes, the security requirements were retained more or less as-is. For compatibility reasons and due to time constraints during the design phase, the UMTS AKA protocol was retained as a component of the EPS AKA protocol.

A main benefit of retaining the UMTS AKA protocol as a component is that one did not have to introduce a new software module on the UICC. Of course, this is also the main drawback, as this rules out more far reaching improvements to the security architecture. In particular, this ruled out using asymmetric public-key based crypto credentials as the basis for subscriber authentication and it ruled out using a Perfect-Forward Secrecy (PFS) based mechanism for key agreement. In retrospect, both these features are going to be needed and it was an opportunity lost not to introduce them in the 4G security architecture.

2) The 4G Security Architecture: The LTE security architecture has a lot in common with 3G security, but with some important changes. Amongst the LTE features are:

- UICC/USIM is retained and required.
- Introduction of full key-deriving key hierarchy.
- Session keys not dependent on re-authentication.
- Auth. master key (K_{ASME}) bounded to VPLMN id.
- New session keys for every handover.
- Separation of user plane and control plane protection.
- Introduction of improved AKA algorithm (EPS AKA).

A welcome change is that backwards compatibility with GSM SIM is prohibited for access to E-UTRAN. UMTS AKA derived security contexts can be used (mapped) to LTE contexts. Figure 3 depicts the EPS key hierarchy, which is very different from the 2G/3G schemes.

The new key derivations take place exclusively outside the UICC/USIM. This makes for a significant departure from previous practices. It also makes the USIM somewhat less significant, given that the mobile equipment (ME) now takes over that functionality.

3) Omissions and Shortcomings: The list of omissions and shortcoming is shorter for LTE, but there are also new types of threats. In a world of smart phones, it is obvious that 128-bit crypto on the access link may count for nothing if the mobile phone is infested with malicious Apps. Likewise, the networks are often hybrid systems, and it is common to have base stations that are 2G/3G/4G compliant. With different security levels and common hardware/software, it is clear that strong 4G protection may easily be offset with weak 2G/3G protection. For 4G this is quite important, as all eNBs will in principle be able to reach all other eNBs.

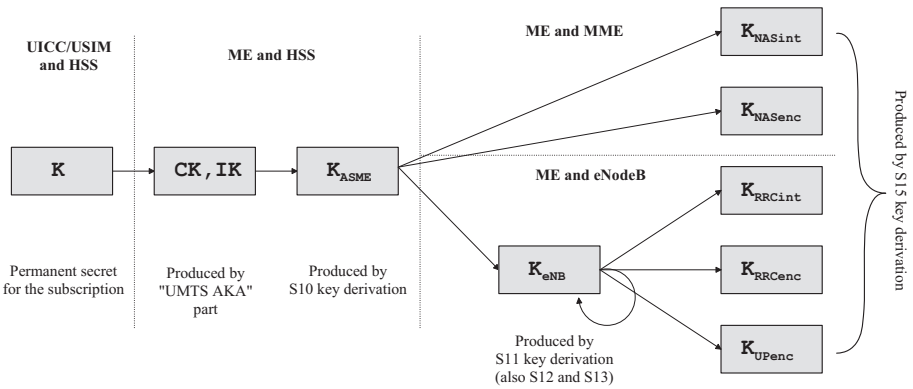


Figure 3: 4G Security: The EPS key hierarchy.

Thus, one compromised eNB can reach all other eNBs in the network segment (which may span the entire operator network). It is also clear that many of the nodes, including the base station (BTS/NB/eNB) may be running commodity operating systems (OS). The chosen OS, likely a Linux variant, may be reasonably secure, but even a high-security OS will have weaknesses and must be properly managed to remain secure. Also, introduction of firewalls and intrusion detection systems will be required for base stations. They have become security sensitive servers and must be handled that way.

Server hardening is a must for all network elements, and even so it is clear that not all attacks can be prevented. This means that prevention alone cannot be a viable future strategy.

The EPS security architecture does require the eNB to be secure, but the specification is not very specific [6]. It also has recommendations on use of firewalls, but the specification is quite vague on this subject too. Altogether the systems cannot be said to be fully specified with respect to security. For a greenfield 4G system, the security may be quite good at what the system provides, but the standard system does not do all it needs to do. Also, it is obvious that the user equipment (UE) must be protected. The UE normally is not owned or controlled by the network operator, but it may still be prudent practice for the HPLMN to offer security software to the users. This is not only to protect the user, which a HPLMN should be interested in anyhow, but also to protect the network as a population of broadband devices could disrupt the access network. Distributed Denial-of-Service (DDoS) attacks would be one possibility [26].

D. Some Architectural Oddities and Vulnerabilities

One puzzling aspect of the 3GPP security architectures is that while identity presentation and entity authentication is fully standardized, there is no authorization mechanisms present. There are of course mechanisms to discriminate subscriber based on the type of subscription, but these schemes are not a feature of the security architecture.

Another aspect to be noted is that the subscriber identity that actually is authenticated, the IMSI, is basically a link layer identifier. Since there is only basic connectivity present at the link layer it may help explain why there never was any built-in authorization scheme in the 3GPP security architecture.

As a high-level observation, we also note that the shared-key basis for authentication and key agreement in GSM, and for that matter in 3G and 4G too, is a liability. One is critically dependent on the security of **a)** the production of the SIM cards and one is critically dependent on the security of the HLR/AuC (or HSS) servers. A related issue is the fact that there is no PFS. That is, given knowledge of the permanent secret key (K or Ki) stored at the SIM/USIM and in the HLR/HSS, it is possible to decrypt every session there ever was with that given subscription. This is so because the other key derivation ingredient, the random challenge ($RAND$), is present in plaintext in the session setup signalling, and thus readily available to the intruder.

Thus, if an intruder records all encrypted calls, it can easily decrypt them all later using the secret key. That makes the secret key a very valuable asset and it represents a huge liability to subscriber privacy. The SIM/USIM authentication secret (Ki or K) is a shared secret and it is embedded in the chip at production time. In many cases the personalization of the smart-card is done by the smart-card manufacturer, in which case the secret key will be forwarded to the HPLMN operator. In this scenario, the trust one must have in the SIM card manufacturer is very high indeed. The required trust in the HPLMN is obviously also very high.

If the core key material had been based on asymmetric crypto, one could have let the SIM card generate the private keys themselves. The embedded key material would then never have been exposed, only the corresponding public part would be copied to the HLR/HSS. Furthermore, if the key material would subsequently be limited to authentication only, one could use it to authenticate Diffie-Hellman key exchanges. These key exchanges provide PFS, and is very much a preferred solution in the possible presence of an intruder with the ability to subvert SIM card production. Due to the Snowden leaks, we now know that a major smart-card manufacturer has indeed been hacked [27].

III. Evolving Security Architecture

A. What Kind of Evolution?

It is, of course, obvious that we are not dealing with biological darwinian evolution. The key components of biological evolution, random mutation and natural selection, are not present as-is. In particular, we have absolutely no reason to

suggest that "random mutation" is involved in system design. Design decisions may appear arbitrary at times, but they are not random. We have no "natural selection" either, but there is a certain level of selection in the sense that solutions that are too weak or useless, in one respect or another, will be a liability to the system. At the extreme one may think about the Heartbleed vulnerability [28] in the OpenSSL software as a selection case. That particular implementation/version of OpenSSL was in some sense put under strong selective pressure to be removed.

So, the closest one comes biological evolution is probably in terms of diversity of implementation and selection of vendors and operators. Diversity, as a means for protection and changing the attack surface, is being explored as a security measure [29]. Another area where one might be tempted to use biologically inspired comparisons, is the arms race between anti-virus products, firewalls and intrusion detection/prevention products and the attacker tools [30]. Overall, however, there seems to be little evidence that security, which is largely standardized, is an arena for darwinian evolution as such.

With biological evolution we essentially have that every generation *must* be competitive in their habitat. It is not possible to skip generations and add entirely new features. With a designed artificial system it may be expected that one may skip intermediate steps, and move directly on to an entirely new feature or function. However, this is not necessarily the case in practice. The new design is often by step-wise refinement, and requirements for backwards compatibility etc. often makes it prudent not to make radical changes. So, in this way, there is indeed quite a few similarities to natural evolution. However, there will also be many small, and sometimes large, breaks with the past. Still, from the vantage point of the overall system, most changes are evolutionary rather than a clear break from the past.

We note that it is sometimes necessary to be more revolutionary to address problems with deep root causes. Security design occasionally comes into this category.

To summarize, the kind of evolution we are dealing with certainly is not Darwinian as such, but the many of the high-level features may yet appear that way. This is particularly true for very large systems, which will tend to behave in a non-deterministic way and where there is no absolute authority to control the system. There may be design authorities and there may be authorities on other aspects, but ultimately these only control a part of the overall system. For instance, with the internet one have multiple authorities control various aspects of the design, the protocols, the address allocation, etc., but nevertheless, none of these authorities have jurisdiction and influence over the overall internet usage as such.

B. Time-line and Security Goals

One immediately obvious observation is that along the system time-line the security goals will evolve with the target system. To maintain isoquant security, the security architecture must provide services that match the developments in threat level and asset values. This means that while the security goals may have long-term validity, the security architecture services must evolve with the greater system context and importantly be able to address new threats. To not improve on the existing security will almost certainly mean that the security level drops. This decay in the system security level is somewhat reminiscent of increasing entropy, and it highlights the need to spend energy/effort just to maintain the current security level.

This also means that if one wants to expand on the scope or provide actual improvements, one "must run at least twice as fast". In Lewis Carrol's "Through the Looking Glass" the Red Queen summarizes this quite nicely.

> "Now, here, you see, it takes all the running you can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!"
>
> *– The Red Queen [31].*

C. The Triad of Protection, Detection and Response

With the 3GPP systems, the protection designed into the security architecture is by and large of a proactive nature. That is, the schemes are focused on the classical security features of entity authentication, data integrity and data confidentiality. These are clearly needed features, and they are provided to ensure that only authenticated and authorized entities will have access to system resources. Additionally, there are concessions to subscriber privacy. This is all well and good, provided that the schemes are sufficiently strong, and that they are consistent and complete.

However, in a large-scale system system, security breaches and incidents will happen with statistical certainty. This is certainly true for the 3GPP systems, which suffers from weak schemes (for GSM in particular), inconsistent security levels and incomplete protection across the system. Obviously then, there is also a need for handling security breaches.

So, we can safely assume that a security architecture must include both proactive and reactive security measures. The proactive (*protection*) measures would be the baseline protection schemes and would include entity authentication, authorization schemes, data confidentiality, data integrity, etc. Traditional server harding would also fall into this category, including the all too familiar administration of security patches and updates.

However, for large system, it is inevitable that there will be security breaches. We can thus postulate that there will be security incidents. With this in mind, it is prudent to be able to handle this. Firstly, one must of course be able to detect that there has been an incident. Obviously then, intrusion- and incident *detection* must be regarded a security architecture requirement.

Of course, detection alone is not enough. One must also handle the detected events, and reactive security measures must be available in the security repertoire. These *response* measures must be fairly flexible since we generally cannot predict what kind of incidents one must be able to handle. To this end, it is both useful and probably necessary to keep human operators in the response arsenal. Humans, while capable of being flexible, are obviously quite slow in the context of an automated attack. Therefore, it seems prudent to have automated response schemes too.

D. Qualitative and Quantitative Aspects

Some security mechanisms are designed to be *secure* as-is. In that sense, they are like a mathematical expression; either

true or false, and usually no middle ground. Now, even if we assume that the base security primitive is indeed fully secure, it is all to often the case that the actual security is compromised by wrongful usage, broken assumptions, erroneous implementation, etc [32].

So, essentially, this means that one cannot rely on a single protection scheme. This is true irrespective of the apparent strength of the primary protection. Thus, there is a need for defense-in-depth. The additional protection schemes may also serve to be backup schemes.

With this in mind, it is likely to be cost-effective to provide defense-in-depth coverage for the assets. That is, provided that these auxiliary schemes are part of an overall design and not ad-hoc schemes bolted onto the architecture. Ad-hoc designs, while they may provide a momentary benefit, is likely to incur future overhead in security maintenance and management. They may also prevent better and more appropriate schemes from being developed and deployed, since they may appear to be effective and efficient (while possibly being neither). Still, we advocate a quantitative defense-in-depth approach to security architecture design.

E. Completeness and Resilience

Defense-in-depth schemes may also serve to add security coverage to areas where the primary schemes may not provide adequate protection. This will reduce the overall vulnerability exposure. Added coverage and multiple layers of security will also provide an opportunity to increase the attack resilience, but we must not be naive here. Only with well-designed defense-in-depth strategies can we hope to achieve an actual improvement. Also, it must be noted that when facing dedicated intruders, simple minded auxiliary security schemes may count for nothing. That is, we must differentiate between protection against advanced persistent threats and protection against unsophisticated attacks by opportunistic intruders [33].

Completeness, whether by means of auxiliary mechanisms or not, is clearly an important goal for a security architecture. This has the implication that for any new system feature or new system assets, one must carefully investigate whether of not the existing security will fully cover it.

Resilience and robustness is likewise very important. For instance, there should be no easy way to disable security schemes by provoking the system into fallback-mode. Fallback modes are all to often a business requirement, but one must take all precautions possible to ensure that an attacker cannot abuse such schemes, or at least to avoid and mitigate serious incidents.

F. Why Low Efficiency May Be a Good Thing

If all security schemes are highly optimized we run the risk of losing flexibility. That is, "high efficiency" protection may be excellent against well-known run-of-the-mill attacks, but they may fail against new and novel attacks.

Generic and flexible protection schemes, may appear a bit "extraneous" and be in some way be less efficient, but they can actually provide protection against new and novel attack. We do not claim that such schemes are necessarily more effective than other schemes, but it is useful to keep in mind the difference between effective and efficient. Therefore, we shall advocate a certain level of security redundancy and diversity, but we note that this must be based in design and that the redundancy and diversity must not be shallow in this respect.

This also means that we must not fall for temptations to deploy lightweight security as the primary protection if there is any reason at all to think that this protection will not be sufficient, consistent or complete in the longer run. As an example, if two-way authentication will be needed in the foreseeable future, then deploying a one-way scheme like the GSM AKA protocol will only hurt the system architecture in the long run. A true two-way authentication protocol may be slightly more expensive to run, but it is future proof and it avoids complications with the need to modify and update it. Likewise, the choice not to use asymmetric cryptographic primitives as the basis for the 3GPP AKA protocol functions may be defended on the grounds that symmetric methods are, computationally, much cheaper to run. However, the symmetric methods are unsuitable for providing PFS, and we know now that lack of PFS is a practical liability [27]. With hindsight, to chose the "low efficiency" asymmetric cryptographic primitives would have been a much better solution than the apparently more efficient symmetric key alternatives.

G. Planned Deprecation

A lot of protection schemes are used even though they are not very secure anymore. Defense-in-depth is one thing, but keeping protection schemes that no longer provide protection is wrong. So, when a crypto primitive is no longer secure one should plan how to deprecate and replace it.

With the 3GPP example systems, it is easy to see that for instance the 64-bit A5 algorithms are no longer future proof. In fact, they are all too weak already. To replace the A5/1 cipher will not be easy, and it is a process that will take considerable time, given that the A5/1 algorithm plays a crucial role in inter-operability for roaming subscribers. However, this only highlight the need to plan ahead and to start the process. When the A5/2 algorithm was deprecated it literally took years before it was officially an algorithm non-grata. And this was for an auxiliary algorithm.

Planned deprecation also implies planned migration, and there may be cut-off dates, etc., involved. This will never be easy to accomplish, but unless one initiates the process early on one should brace for the impact when change is forced upon the system. With this in mind, one should always include a "best before" date on all security components and mechanisms. This was actually done for the KASUMI cipher [34]. The actual statement made in the evaluation report was that the algorithm should be reviewed every five years to verify the security and usability [35]. In practice, this meant that KASUMI was deemed safe enough for 3G security but unsuitable for 4G security. It also meant that 3GPP commissioned the development of an alternative to KASUMI, the SNOW-3G algorithm [36].

The lesson is that one should plan for the schemes eventual deprecation during the design process when the scheme is included in the security architecture.

H. Facilitating Secure Migration

As stated above, deprecation of a security scheme will tend to imply migration to a new scheme. When this is the case, it is of course imperative that the migration process is secure. This can be quite difficult to achieve, since it is more than likely that the old and new scheme must co-exits for a considerable time. This again implies a capability to securely negotiate the right scheme. It also implies a well-though out migration plan and security policies that matches this.

As a fact of life, one may also need fall-back from a newer security scheme to an older scheme. This usually implies going to a lower security level. The triggering conditions may include incompatibilities between the negotiating entities, but whatever the fall-back decision is based on, one must make sure that an intruder cannot trigger this condition too easily. We casually observe that a legitimate system entity may also be an intruder. Fall-back options are messy and very hard to make secure. To allow them means accepting higher risks, and fall-back solutions should certainly be monitored and they should certainly have their best-before dates. It is of course also essential that they are captured in the security policies.

The requirement for secure negotiation of security schemes must necessarily mean that there is a security basis to facilitate the negotiation. This security basis must be valid in a long-term perspective and it must be rock solid and fully trusted. Efficiency is not a primary requirement here, and it is akin to a root certificate in a PKI system. The root certificate may have excessive key length and use computationally inefficient algorithms, but this does not matter since it is used infrequently and since rock solid security is the only real imperative.

I. Mitigation and Recovery

Mitigation and recovery is in many ways part of the *response* requirement. However, we want to make it explicit that schemes that exclusively facilitate mitigation may have a place in the system. These schemes merely reduce the impact of an incident, but that may be a worthwhile goal and it may also be a cost-effective option.

Recovery schemes will obviously also be needed. These are after-the-fact schemes that simply aims at restoring operation after an incident. Needless to say, initiating a recovery operation must be subject to authorization.

J. The Scalability War

The classical Dolev-Yao intruder model is not the most realistic intruder model [37]. Real intruder will use any available means (subversion, physical intrusion, tricking the principals), ultimately being as powerful as a Dolev-Yao intruder. An National Security Agency (NSA) type of intruder will obviously also use the legal procedures to get access to systems. There is a reasonably body of papers detailing various intruder models, but suffice to say that a modern CI system must be able to handle **all** types of intruders. Furthermore, the CI system, inevitably exposed by having an internet presence, must face the prospect of distributed attacks. Distributed Denial-of-Service (DDoS) attacks are not new, and they may also be initiated over wireless connections [26, 38]. Other types of distributed attacks are also possible, and they may actually use DDoS attacks as a means to trigger error conditions, which then are exploited.

This inherently means that the system *must* have efficient as well as effective protection, and that mechanisms that do not scale well, compared to intruder capabilities, will be doomed to fail in the long run.

Our assumptions related to scalability and efficiency:

1) Security scalability will be a major concern.
2) Efficiency is highly important.
3) Effectiveness is imperative for core mechanism.
4) Auxiliary defense-in-depth solution are needed.
5) Avoid specific-attack measures if at all possible.
6) Security management must scale well.

See [39] for some considerations concerning scalability in general in the world-wide web context.

Assumptions three and four are apparently somewhat at odds, but in the end assumption three can be supported given that these means are complementary and cost-effective. See also considerations about the economy of attacks and defenses outlined in [33], This indicates that for broad sweeping attacks, even quite weak mechanisms may successfully thwart the attacks. Measures that are only effective for one specific attack should generally be avoided.

One must be able to handle a multitude of opportunistic, but probably not too capable, intruder and one must provide reasonable protection against capable intruders. There is also a significant difference in those attacks that scale effortlessly and those that do not. Defense schemes whose sole purpose is to increase the attack cost may therefore have a justification.

IV. Why Change the Security Architecture?

The short answer is that we need to change the security architecture because some of the premises for the original security architecture have changed. A slightly longer answer would revolve around the following aspects.

A. High-level change triggers

There are many high-level change triggers, amongst others:

- *Changes to the assets of the system.*
  This could include changes to the value of the existing assets, inclusion of new assets or removal of assets.
- *Changes in the threats towards the assets.*
  This includes assets exposure, new intruders, new intruder capabilities. For new assets it could also include missing or mismatched protection.
- *Changes to the system context.*
  The system may initially have played a limited role, but may have evolved into something more.

The engineering aspects of security design and implementation are not new [40], but likewise it is not exactly new either that there may often be a mismatch between the design requirements and the real-world threats and needs [32]. For the 3GPP systems, it is quite clear that the financial value of a network operation has increased sharply during the lifetime of the 3GPP systems. That is, there are many orders

more of subscribers than there originally were. The assets have similarly evolved such, and not surprisingly, the threats towards the systems have changed substantially over the years.

B. Evolution aspects

Large-scale long-lived systems cannot remain as static objects for long. Instead, they must be dynamic and adapt to changing environments. This is true of the 3GPP systems too. A network operator that only provide speech and short messages will not be as attractive as operators with a more complete set of services. Price will to some extent influence this, but then one may see a lower relative price as a change to the value of the assets, and as such it is in some sense an adjustment to a changing environment.

- *Evolving Target System.*
  If the target system changes, then this will likely affect the security architecture. Still, the nature of the change may be such that it does not trigger a need for updating the security architecture.
- *Evolving Security Architecture.*
  The security architecture may need updates and modifications due to external circumstances, or even completion of planned features that were not initially fully specified. Changes in the threats towards the assets, the exposure of the assets, and the number of users will also affect the system. It could also involve changing trust-relationships and changes to value of the assets. All these are at play with the 3GPP systems.
- *Security Evolution History.*
  An evolving system is obviously a product of its history. Decisions taken during the design of GSM still have an impact on LTE. For instance, the basic identity presentation scheme essentially remains the same for LTE as for GSM [41, 42].
- *Societal Impact.*
  When a system reaches certain thresholds it will take on a new role. It enters a state of criticality to society and will become an object of regulatory interest. The critical infrastructure (CI) requirements will focus on system survival and service availability rather than security and privacy for the individual.
- *Privacy.*
  Privacy requirements may not have mattered too much for a small system with few users back in the early 1990ties. Today, privacy requirements are often mandated by laws and regulations [43].

V. Assumptions Regarding Systems, Security and Cryptographic Characteristics

The following set of assumptions not all be true for all systems, but we advocate assuming that they are true.

Some of the assumptions are relatively self-evident in nature, while others may appear less justified. Nevertheless, the value of these assumptions is more as guidelines to a design process than as propositions that must be defended.

A. Assumptions about Successful Systems

We assume that when people start to design a system they intend it to be successful. Thus, they must take the above into account in their design. Our high-level assumptions about a successful system:

1) It will outlive its intended lifetime (and design).
2) It will have many more users then originally intended.
3) It will need to scale its services cost-effectively.
4) It will become highly valuable (many/valuable assets).
5) It will outlive its base technologies.
6) It may become a critical system (company, organization).
7) It may become a critical infrastructure (society-at-large).
8) It will spawn unsuccessful branches/features.
9) It will have to deal with multi-vendor cases.
10) It will need to operate with multiple releases in place.
11) It must encompass all of operations & maintenance too.
12) It will be subject to regulatory interventions.

B. Assumptions about System Security

Our assumptions about a long-lived security architecture:

1) The assets will change (value/number/types).
2) The principal parties will change and multiply.
3) The threats will change.
4) Trust models will fail (and/or become outdated).
5) Trust will be betrayed.
6) Risk evaluations will be outdated.
7) Weaknesses, vulnerabilities and exposure will change.
8) Intruders will become more powerful and proliferate.
9) Attacks will only be better over time.
10) There will be security incidents.
11) Scalability in security mechanisms will be decisive.
12) No single security scheme or approach will suffice.
13) Effective and efficient defense-in-depth will be needed.
14) Pro-active security protection will not be sufficient.
15) Re-active security will be very important.
16) Ability to handle large incidents will be required.
17) Deprecation of security schemes must be built-in.
18) Secure fall-back must be supported (but not trusted).
19) Security negotiation must be built-in.
20) Mitigation and recovery must be supported.
21) Pervasive resilience and robustness is required.
22) Autonomous response will become important.
23) There will be security architecture omissions.
24) There will be security issues (multi-vendor).
25) There will be security issues (multi-release).
26) Fixing minor security wholes can take a very long time.
27) Fixing the security architecture may take years.
28) Security management will be crucial.
29) Security configuration management is crucial.
30) Security migration methods should be built-in.
31) Security policies will be inadequate and incomplete.
32) Security policies will be outdated.
33) Privacy will become ever more important.

This list of assumptions should not be read as a definitive or authoritative list, but rather as a starting point.

C. Assumptions about Cryptographic Solutions

Our assumptions related to cryptographic solutions:

1) The cryptographic base functions must be future-proof.
2) Cryptographic primitives will be broken (or become too weak).
3) Key sizes will be changed.

4) Security protocols will be broken (or become too weak).
5) Cryptographic parameters will need to be negotiated (securely).
6) Cryptographic primitives will need to be revoked.
7) Implementations will contain weaknesses.
8) Management of cryptographic elements will be crucial.

It is clear that the basic boot-strapping fundament must be very solid. This minimal base is what you will depend on if you need to boot-strap new security solution and new cryptographic primitives in the rest of the security architecture. It needs to contain enough to support boot-strapping and it needs to be future-proof. Efficiency is *not* a main priority here.

VI. Too Big to Fail?

Even very large systems can, and almost certainly will, fail at some point in time. Consider the collapse of the Soviet Union [44], the bankruptcy of the Lehman Brothers Holdings bank [45] or for that matter the rise and fall of the AltaVista search engine [46]. In all three cases, these were large and powerful entities in in their respective domains.

System failure may be temporal, partial or spatially confined. It may also be permanent, complete and global. In this section, we take a look at some of the factors that may contribute to failure. Much of this section will be conjecture. The purpose is not to derive grand conclusions, but rather attempts at understanding a little more about some of the factors that come into play. That, and pointing to areas were we need more research.

A. Evolution and Architectural Decay

Big systems are complex entities, and security architectures no less so. There is, at least initially, a high degree of structure in how the architecture is organized. The complexity one finds will therefore tend be be necessary complexity.

Evolution implies changes, and unless meticulously executed, the changes will complicate the architecture. Some of the complexity will then tend to be a product of the change process itself. The complexity will increase, but the structure may actually be less clear. In short, there will be increased entropy. In thermodynamics, the entropy increases due to random changes to a system with a high degree of structure. The changes will have (with statistical certainty) a higher likelihood of distorting the structure than improving it.

Designed evolution is not of course random by nature. Still, with many different and competing requirements, it is to be expected that some of the design decisions will be sub-optimal or counter-productive with respect to maintaining structural consistency. In a highly organized system architecture, this will inevitably lead to a less structured, less coherent and less consistent system over time.

Requirements for backwards compatibility will complicate matters, and this is almost always something that will lead to less structure and/or less consistency. Given that a designed evolution is not random, one may expect that many of the design decision will actually improve the structure. Thus, there is a counter action to the architectural decay.

For the 3GPP systems, it is essential that backwards compatibility with older and insecure security schemes is deprecated to avoid architectural decay. To some extent this is happening, and the fact that the GSM AKA protocol cannot be used for authentication and key agreement in LTE is a sign of that. However, it is essential that the rate of deprecations and obsolesce does not lag too far behind the rate innovation and new schemes.

B. Black Swans

The theory of black swan events describes black swans as something completely unexpected, yet with hindsight it appears much more predictable. The concept is derived from the fact that prior to discovering Australia, Europeans simply could not envisage something like a black swan.

The concept has been popularized by Nassim N. Taleb [47, 48] and are associated with financial events. The Lehman Brothers collapse was a black swan event in this sense. Taleb defines black swan events this way:

1) The event is a total surprise.
2) The event has large and even severe effect.
3) The event was later, with hindsight, seen as predictable.

In terms of the financial systems background, Taleb attributes 1) to a failure of understanding the statistics properly. This in part is due to not understanding the nature of randomness and not understanding that statistical distributions simply are not well defined for singular or very infrequent events. That is, you cannot reliably determine the confidence interval, deviation or frequency of a class of events with little or no historical data. This, in effect, means that you cannot rely on statistics to predict those events since you do not even know the distribution. As for point no. 3, it usually seems clear in retrospect that the risk was always there. Given new knowledge, it will even seem obvious that the black swan event occurred. This is of course the benefit of hindsight, and can to a large degree be attributed to what Taleb describes as our inability to acknowledge the role of randomness. When the result is known then it is indeed no longer a surprise.

In terms of security and security architectures a black swan event would be something that simply is not captured at all. This could be due to the magnitude of the event(s) or down to the combination of events.

It could also be down to events that simply are not captured by the system model or down to our lack of understanding what the real system threats are. This is in particular a risk for evolved systems. In our case study object, the 3GPP systems have a number of both standardized and non-standardized security measures. These have evolved over the years, and have grown to address most of the perceived and experienced threats. But, at the same time the exposure of vulnerabilities/weaknesses change and the attack surface probably increases, with or without being acknowledged. Most of the security measures are inadequate in the sense that they do not stop or fully address the threats, but they do impede and/or mitigate the threats such that the risk is low (or believed to be low).

Part of the security risk problem is that we generally do not understand threats or risks very well. People, even professionals, are not good at foreseing which threats are realistic and not, we fail to foresee impact and we do not

really understand scalability. The last part is important, since scalability is what ultimately may break a system. A successful attack on 10,000 cellular subscribers is bad in many ways, but if an attack is limited to that level it will not pose a threat to the cellular networks as such.

In the same sense that we do not fully understand how attacks scale, we do not really know how defense mechanisms scale either. Even a weak defense-in-depth auxiliary scheme may be effective on a system level. It may not be effective against advanced persistent threats, but could fend off broad-scale automated attacks. Even system diversity options that were not intended to be security mechanisms may contribute here since they alter the attack surface [29, 33].

C. Why Don't We Listen to Warnings

In Greek mythology, Cassandra was a daughter of Priam, the King of Troy. Cassandra was a very beautiful lady, and she attracted the attention of Apollo. He provided her with the gift of prophecy, but got vengeful when Cassandra refused his romantic advances. He then cursed her so that nobody would believe her warnings. So, while she could foretell future events, she had no way of altering the events or convince others about future perils. This is why her warnings about the Trojan horse went unheeded.

Warnings about inadequate security or of new and potentially devastating threats come and go. There are so many security inadequacies and yet they do not seem to cause severe problems. This apparent paradox is explored in the paper "Where do all the attacks go?" [33], and part of the answer seems to be that most attacks are unsophisticated and opportunistic by nature. The intruder go for the low hanging fruit and do not necessarily target a specific system or host.

In the 3GPP realm, we have examples of appallingly weak security and yet the protection somehow seems adequate for the purpose. The GSM authentication is one-way only, the encryption is only 64-bit wide, there is no integrity protection, and yet GSM seems adequately safe for what it is. Still, there is of course a tipping point there somewhere, where attacks get practical (this has happened) and where the cost of doing so gets sufficiently low to allow everyone to do it.

There are many people warning about the GSM weaknesses [13–15, 23, 27, 49], and one might even describe this as a chorus of Cassandras [50]. However, if a catastrophe does not occur on schedule, we tend to discount the messenger. Cry wolf to many times and people get tired of the message and the messenger.

The complexity of a modern critical ICT system is daunting, and we cannot blame even the system architects for not fully understanding it. Security is in many way even harder to understand as it is fundamentally about missing, incomplete or inadequate functionality in a given context (and where the context is continually subject to change). So, top level management must learn to live with false alarms (noise) and will have to dismiss them regularly. This, of course, makes it only harder for an important warning (signal) to get through.

That is, security architects must learn to live with warnings not being heeded.

D. Unprepared and Unaware of It

In the paper "Unskilled and Unaware of It: How Difficulties in Recognizing One's Own Incompetence Lead to Inflated Self-Assessments" [51], the authors investigates the ability of people to assess their own competence. While this may be culturally dependent and while it may not apply under all circumstances, the paper highlight the fact that incompetent persons do not necessarily recognize their incompetence. Since they know so little they have no way of knowing what they do not know.

Curiously enough, the opposite seems also to be true. The best skilled people often do not recognize how good they are, since they tend to compare themselves to other highly qualified people. The end result is that they see themselves as more average than they really are.

For large security architectures like the 3GPP system, we have that there is no single person or individual company that has full design authority of the system. We have, as noted earlier, the 3GPP system represents a clear case of contribution-driven design-by-committee regime. The individuals or these committees may or may not be unskilled or unprepared, but one may speculate sometimes if the organization of the specification work is such that the design appear to have been done by an unskilled architect. There may certainly be awareness of this on the level of the participating individuals, but it is not at all clear that there is awareness on the organizational level.

In terms of security, we have that it is generally very difficult to correctly assess threats and risks. It is also very difficult to assess what good security really is. The Heartbleed incident comes to mind again [28]. This was not a design flaw as such, but implementation errors do happen and one must be prepared to handle incidents whatever the cause. Too many organizations were unprepared for something like this, and moreover, they seemed unaware of their unpreparedness.

This will easily lead to situations where one invest unproportionate time and money on features that may in the end only prove a bare minimum of protection. Those schemes may be important as-is, but one the whole the balance is uneven when considering where time and money was spent. The problem is even more acute for security architectures than for single schemes. So, it is conceivable that we face a situation where the very best experts do not assert themselves as they should have and where people without real expertise may exert considerable influence.

The unskilled/unprepared paradigm may also accentuate the so-called "Bikeshedding problem".

E. Painting Bikesheds

Parkinson's "law of triviality" is commonly referred to as bikeshedding. The law is associated with Parkinson's 1957 observation that organizations give disproportionate weight to trivial issues [52]. The "law" emerges from a case-study of a committee whose job it was to approve plans for a nuclear power plant. The committee spent a lot of its time on trivial and unimportant issues. These issues were easy-to-grasp and did not require insight, preparation or deep understanding.

Amongst the items discussed was material choice for the material to be use for the staff bike-shed. At the same time,

the committee almost neglected discussing the proposed design of the nuclear power plant itself. Of course, that design was complex and it would have taken a real effort to comment on it with insight,

The term "bikeshedding" was further popularized with the email posting of "A bike shed (any colour will do) on greener grass..." to the FreeBSD mailing list [53]. The author, Poul-Henning Kamp, cites the discussions about the updating of a minor function in FreeBSD, and then goes on to explain that:

> A bike shed on the other hand. Anyone can build one of those over a weekend, and still have time to watch the game on TV. So no matter how well prepared, no matter how reasonable you are with your proposal, somebody will seize the chance to show that he is doing his job, that he is paying attention, that he is *here*.

The bikeshedding problem has of coursed been recognized and there have been attempts to mitigate the effect. Many large-scale system design efforts have an individual as the ultimate arbiter for cases like this. These individual have authority to make final decisions and to stop useless discussions about unimportant features or minor aspects. For instance, in the Linux world, the original designer, Linus Torvalds, have more or less absolute authority over what code is included in the Linux kernel. For the programming language Python we have a similar situation, where the original creator, Guido van Rossum, is appointed Benevolent Dictator For Life (BDFL). We note that an organization like the 3GPP does not really have a any person with ultimate design authority. The closest one comes is the plenary high-level design forums, but these do no give directions or guidance needed to avoid bikeshedding as such.

Given that security and security architectures are indeed very complex entities, we should not be surprised to see a fair amount of bikeshedding here too. This will add to the noise and divert attention of decision makers, so it is important that we are aware of this effect.

F. Inverse Bikeshedding

Complementary to the "law of triviality", we have another phenomenon that sometimes surface, namely what we term the "inverse bikeshedding" phenomenon. This phenomenon is more or less the opposite of "bikesheeding", and here we have an obsession with attention to highly complex technical details at the cost of ignoring the larger picture.

This is the kind of trap that very clever specialists may fall into. For instance, in the Crypto Forum Research Group (CRFG) associated with the Internet Engineering Task Force (IETF), one could witness heated debate over the elliptic curves used in cryptographic primitives and signature algorithms in spring 2015 [54]. Now, within the context of CFRG it makes sense to focus on cryptographic detail, but it seems that many of the participants fail to see that the actual adoption of an elliptic curve is unlikely to have any relation to the minor differences between the discussed alternatives. Local to the group, the discussion is valid and on topic, within the overall IETF context, the discussions are acceptable if they quickly lead to actionable decisions. Should the discussions not lead to timely and relevant conclusions, then they would appear to be "inverse bikeshedding" activates instead.

It is a leadership challenge to avoid "bikeshedding" and "inverse bikeshedding" activates, and the role of the group chairs is instrumental to reach timely conclusions.

G. Security Theater

After major incidents there is a need to be seen to "do something". Thus, not only is there a strong incentive to point out who the bad guys are, but also to come up with measures that appear to counteract the newly discovered (or newly acknowledged) threats. The 2014 hacking of Sony [55] emphasizes the hunt for a culprit. In other cases, we see disproportionate and even completely misguides responses, and we saw quite a lot of that in the wake of the September 11 2001 attacks on the Twin Towers. Bruce Schneier is generally credited as having coined the term security theater and in the book "Beyond Fear" [56] he elucidates the concept. The concept is further refined in [57].

Security theater is not all bad. In particularly, it may offset over-reactions to incidents and allow business to continue as usual where fear would otherwise dominate too much. One may view this part of the 9/11 response as a measure against fear. Since to instill fear is a major goal of terrorists, the illusion of security theater can be seen as a counter-measure to illogical fear of terrorism.

However, security theater is also wrong. Part of this problem is a that one tends to end up with a lot of attention to strengthen unimportant features, often combined with a strong and narrow focus on details. This will not improve actual security very much, and it will in many ways be counterproductive as it diverts resources and attention onto trivial matters. As such it will foster more bikeshedding and a false sense of security.

H. False Security and Cargo Cult Security

Security theater may over time develop into the more elaborate *cargo cult security* type of deception. Then the main functions and mechanisms may all be there (or mimicked closely), but with some vital part missing or done completely wrong. Cargo cultism is defined by "perfect form", but it simply does not work as intended. Feynman has an amusing description of "cargo cult science" that nicely illustrates the principles [58]. Since security can be very difficult to get right and to verify, cargo cult security may look like the real deal.

Within the 3GPP security architecture one would be hard pressed to find cargo cult security, but if one looks at the wider picture with deployed networks one may find both false security and even cargo cult security.

It is worth noting that those that champion cargo cult security may not recognize that they do so. Either way, cargo cult security is antithetical to real security and may lead to a false sense of security. To do something right is not enough, one must also do the right thing.

I. Trust and The Tragedy of the Commons

The article "The Tragedy of the Commons" [59] is often cited and is an example of a game theoretic problem in which individuals acting independently and rationally according to each's self-interest, behave contrary to the best interests of

the whole group by depleting some common resource. The commons in questions was originally about unregulated grazing rights on common land, but it has application for any common resource accessible to many parties. The problem is an optimization problem, in which the best long-term strategy would be for the individuals to behave cooperatively. However, if enough individuals defect, then it no longer pays out to stay loyal and the best strategy would be to defect.

Security and security architectures are not really a commons resource, and the tragedy of the commons does not necessarily play a direct part here. However, it can be seen to play a part if anti-social attitudes and downright theft becomes the norm for a large enough subset of the population of the users. This will seriously affect the trust climate and systems and societies needs trust to thrive [60]. Without trust, many if not most, transactions would be much more cumbersome and much less effective. There is a soft side to trust and there if a hard side to trust. The hard side consist of methods for enforcing trust and requiring trustworthiness. Security procedures will be amongst the more important ones in the arsenal of hard trust.

So, if soft trust is not seen to pay off, one will often react with tougher hard trust requirements. This could be well justified, but increasing the security level will often have consequences for usability and the transaction costs. This will ultimately have an impact on how efficient the system is.

J. The Somebody Else's Problem and Bystander Apathy

Somewhat related to the tragedy of the commons problem, we have the so-called "Somebody Else's Problem (SEP)". The SEP is humorously explained in the novel "Life, the Universe and Everything" in the "The Hitchhiker's Guide to the Galaxy" books by Douglas Adams [61].

> An SEP is something we can't see, or don't see, or our brain doesn't let us see, because we think that it's somebody else's problem....
> The brain just edits it out, it's like a blind spot. If you look at it directly you won't see it unless you know precisely what it is. Your only hope is to catch it by surprise out of the corner of your eye.

The SEP is understood as a phycological perception problem, in which nobody feels responsible for addressing a particular problem because it is seen as somebody else's problem. The SEP effect is not of course limited to security and security architectures, but it does certainly have an effect here too and it may effectively prevent obvious weaknesses and threats from being addressed.

The somebody else's problem is related to the so-called *unresponsive bystander problem*, named after the case of the killing of Kitty Genovese, in which there were no less than 38 witnesses to the stabbing [62]. As it turns out, people become less responsive to a problem if they do not perceive it as their problem. The feeling of ownership of the problem is substantially weakened when there are other persons present. So much so, that the feeling of responsibility seems to vanish more or less completely when more than four persons are present [63].

Security problems may sometimes come in the SEP/Bystander category. The problem is exacerbated by the fact that sometimes one cannot easily place the responsibility for a problem. Is the given issue a design problem, is it an implementation problem, or is a deployment and configuration problem? This problem is localized in the sense that the SEP apathy may affect anyone, and holistic in its negative effect on the overall security architecture.

Clarifying responsibilities will certainly help with the SEP/Bystander problem, as it reduces the number possibly responsible "bystanders". Organizations like the IETF and the 3GPP have at least some shallow mechanisms in place to that may address the SEP/Bystander problem. There are styleguides on standards documents dictating separate security sections and there are requirements to check for security impacts on new functionality, etc. Still, the SEP/Bystander problem requires organizational awareness to correct it, and it is a leadership responsibility to see that the problem is addressed. We note that there are elected officers and chairpersons in for instance the 3GPP and the IETF.

K. Inefficient Enforcements and Susceptible Parts

The "The Byzantine Generals Problem" [64, 65] is a well-known problem generally concerned with fault tolerance. The Byzantine problem has its background from the Byzantine military, in which each division is controlled by a general. The generals communicate by messengers. Some of the generals will be traitors. How then, in the presence of traitors, can the loyal generals adopt a good plan? The question can be loosely translated into "How many components can we tolerate to fail".

Component failure in large-scale system is not a matter of if, but when and how often. Accidental component failure is one thing, but component failure due to attacks and subversion is another and more serious problem. In a large system, no matter how good the security architecture may be, we will experience weak links. Some of those will be substantially weaker than what the security architecture mandates. That is, we have *inefficient enforcement* of security requirements. Or, of course, the requirements itself may be missing. There are a huge number of reasons for this being so, but rest assure that this condition will affect a certain number of the population of nodes, components and parts in the overall system.

Single failures are probably not problematic at the system level, but we will likely have Byzantine conditions in the system. The security version of the Byzantine problem goes beyond the fault tolerance version in that the problem is more severe. The common part is that below a certain threshold the system cannot be made reliable or secure anymore.

The only viable way to handle this problem is to have good and well rounded detection and response mechanisms in place, together with various redundancy schemes. Good redundancy schemes will improve the system resilience and robustness, but one must ensure that the redundancy is effective and this requires verifying the appropriateness of the redundancy schemes regularly. For instance, if flooding is the problem then placing a redundant server in the same location as the main server is unlikely to be a good solution. In the 3GPP systems, redundancy is not normally part of the design. At best, one has catered for the possibility. This means that redundancy must be part of the implementation, and part of the operation and maintenance of the deployed system.

L. Thresholds and Tipping Points

The Byzantine Generals Problem does point to the fact that there are thresholds, beyond which the system breaks down. Too many traitors amongst the generals, and the decisions process can be effectively subverted. For a large systems, local breakdowns or temporally confined outages are something which one routinely will have to handle. These will not break the system, but they will add to the burden and complexity of maintaining the system.

However, there will inevitably also be breakdowns that cannot be handled so easily. There will be global thresholds and much like in chaos theory, the system may actually look reasonable stable while it is located within its basin of stability [66]. In chaos theory, nonlinearity effectively prevents long-term predictions, even though the system may be mathematically deterministic in nature. That essentially implies, if the system, for one reason or another, strays outside the basin of stability, the outcome will largely be completely unpredictable. This being said, there are also progress towards anticipating these critical transitions [67]. With or without anticipation, there may be system-wide tipping points, after which there are no obvious recovery anymore. That is, recent research points out that recovery, at least in living systems, seems to be related to the "distance" from the tipping point [68]. This provides a glimmer of hope.

Finally, we note that the system may be seem very stable and appear to be highly resilient while within its basin of stability (*basin of attraction*).

These insights are not easily absorbed in security architecture designs, but one lesson seems to be that while one may be unable to prevent such incidents, one may be able to respond to them. This type of "Black Swan" incident would be very hard to predict and the response would be equally hard prescribe a priori. This can be seen as an argument for redundancy and deep diversity in terms of mechanisms available in the response repertoire. It may also be construed as an argument in favour of keeping skilled humans in the loop. It is an argument for emergency response rehearsals and preparedness in general.

M. Unknown Unknowns

The phrase "unkown unknowns" comes from Donald Rumsfeld's perhaps most famous statement while serving as George W. Bush's secretary of defense:

> As we know, there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns the ones we dont know we dont know.
>
> *Donald Rumsfield, February 2002*

Politics aside, the statement does point to a truth about what we can prepare ourselves for. A security architecture should obviously have measures for handling threats and attacks we know about. These types of threats and attacks are so predictable that we can plan for them with pro-active baseline security measures.

There are also those unknowns that we know may be there. These are foreseeable and while we do not exactly know how they may materialize, we know enough to plan for handling them. Pro-active baseline security will still play a role, but we also need adaptability and must rely more on re-active schemes. These will require detect-and-respond capabilities.

The "unknown unknowns" are a tougher lot to handle. Part of the problem is that we do not know what to look for. Strategies depending on *detection* as a key element will have to be very flexible in order to catch these type of threats. Humans are generally better at recognizing novel threats than automated systems are, and so putting humans in the *detection* loop seems prudent practice. Still, our ability to distinguish between chaotic data and random data can be questioned, and this will surely impact our ability to detect true patterns. Here, automated systems will need to play an important part. Humans are also notoriously slow when compared to machines, and so an attack could be executed and completed well before a human would be able to respond. All-out attacks will of course be detectable, but by then it may be too late.

Digital one-off pinpoint attacks may also be virtually undetectable in the sense that whatever pattern there were would drown out in the noise of normal behaviour. However, these attacks would not normally constitute an attack on the overall system, and may as such be tolerable (from a system point of view).

N. Guaranteed Eventual Failure

Empires come and go. The Roman Empire fell. The British Empire fell. The Soviet Union fell. They seem all to fall in the long term.

Big corporations come and go. If one investigate the destiny of Fortune 500 companies, it is apparent that even large corporation come and go with a fairly high frequency. A comparison of the destiny of Fortune 500 firms in 1955 vs. 2011 shows that 87 percent are gone (from the list) [69]. Furthermore, the life expectancy of Fortune 500 companies have declined from 75 years and down to less than 15 years.

We have little reason to assume that large-scale ICT system will endure indefinitely. This implies that the system will somehow fail. So, as a postulate, we shall claim that all system will eventually fail. It will be interesting to learn if such failure will be related to how the security architecture fares. We know that the weaknesses of the security in the 1G cellular systems was a contributing factor in the demise of those technologies, but it also clear that GSM and other 2G technologies would have replaced the 1G system irrespective of the merits of the security architecture.

What we do not know very much about is how these systems will fail. As long as the systems are critical ICT infrastructures, it may seem that they will either endure or fail in a disruptive collapse. Of course, as technology progresses, the failure may simply be a slow, but probably accelerating, decline into obsoleteness.

VII. Lessons Learned

A. Assets, Nodes, Entities, Threats and Intruders

Make sure that one has an updated inventory of a system assets, the network elements and nodes, the participating parties/entities,the threats and the most likely would-be intruders.

This is a detaining task and it must be done regularly. We advocate using tools for this purpose, and the Microsoft Threat Modeling Tool is a good practical alternative [70].

B. Requirements and Policies

Threat modeling alone does not solve our problems, but it may help significantly in identifying the consistent parts of the security architecture. The threat modeling tool mention above will, for instance, effortlessly allow requirements to be distilled and appropriate protection to be proposed. Threat modelling may, for instance, also make the requirements clearer and it may also help in defining the security policies.

C. Verify Assumptions

One must verify assumption about the system and the security periodically or when there are substantial changes to the system. That is, an audit is called for to verify assumptions about the assets, the principal entities, trust relationships, etc.

Security policies must be adapted according to changes to the assumptions. This is a process oriented task that must take place both for the design phase and for the deployed system(s).

We want to highlight that even non-technical aspects such as trust must be carefully reviewed. We also want to point out the difference between trust and trustworthy. The fact that one trust someone's intentions does not imply that one can trust their ability to behave according to intention. Thus, we must assure ourselves that our partners are trustworthy. The trust assumption should of course be explicit and concrete.

D. Rock Solid Bootstrapping Security

There needs to be a rock solid fundament that will be secure for the foreseeable future. The smart-card has served this purpose in the 3GPP systems on the subscriber side. The smart-card is not tamper-proof, but it has successfully served as a high-trust platform.

That being said, a recently leaked document from the British Government Communications Headquarters (GCHQ), shows that NSA/GCHQ have at least hacked one of the major SIM card manufacturers. The target company, Gemalto, is a large multinational firm based in the Netherlands that produces in the order of 2 billion SIM cards a year. The leak is part of the Snowden files and is published at The Intercept [27]. It is worth to note that Gemalto probably is one of the most security conscious companies out there, but they obviously were not impenetrable in the end. This may in itself be a lesson.

The leak does not weaken our requirement for a rock solid bootstrapping base, but it highlight the need for ensuring that the trust in the base is warranted, enforced and validated.

E. Planned Deprecations

A scalable and evolving system must be able to handle deprecation of almost all cryptographic algorithm, security protocols and security services. The deprecation, needless to say, must be conducted in a secure manner. Backwards compatibility requirements and fallback solutions must be handled in a secure way.

F. Negotiable and Adaptable

Given that one must plan for deprecation of security features/services, one must also plan how to negotiate new features/services. This feature must be built-in and have high assurance. Adaptation may be necessary to account for local requirements, but is vital that adaptations must be fully compliant with a well-defined security policy.

G. Proactive & Reactive Security

Basic security functionality to identify and authenticate principals and entities is necessary, but not sufficient. Adding authorization, protected storage and protect communication is also necessary, but still not sufficient. More may be added, but in the end it is impossible to fully secure the system. This means that one must handle and deal with incidents. Therefore, there is a clear need for intrusion detection and response systems, to deploy firewalls, anti-virus protection, secure backups, secure audit trails etc. The reactive measures must be included in the overall system security plans and subject to revisions as need be.

H. Stability, Resilience and Recovery

System integrity is imperative to ensure a stable and resilient system. System integrity is a system-level characteristic and does not preclude partial or local failures. What is imperative is to prevent the failures to scale. Failures, whether man-made intentional or unintentional, cannot entirely be prevented. Procedures that support mitigation and recovery must be an integral part of the overall system security plan.

I. Configuration Management

Proper planned configuration management, which must include security functionality, is an absolute necessity.

J. Memoryless Security

Security will fail, and then it is prudent that the impact is contained. This speaks strongly in favor of security protocols and crypto systems that are "memoryless". That is, perfect forward secrecy (PFS) should be included as a major principle.

K. Privacy Matters

Privacy is one feature that must be accounted for in all systems that include human users or any kind of data pertaining to humans. This must be planned for from the design phase and handled in all phases of system deployment.

Privacy is, however, also a difficult concept and largely a culturally dependent trait. What can be expect to keep private, and not the least, from whom do we keep information private. Nevertheless, whatever privacy level we decide on, one should ensure that it is credibly maintained.

VIII. Discussion

A. Evolution

In this paper, we have outlined the 3GPP security architecture as it has evolved over more than 25 years. From being an auxiliary service for the few, it has grown to literally cater to billions of subscribers, and the number and types of services provided has changed dramatically over the years. The use-patterns of these systems has changed as well. All in all, there has been a complete transformation of almost all aspects of these systems. During this process, the security architecture has evolved with the system and the changing system context, though not without some noticeable failures and a growing number of security problems.

We have argued that to achieve scalable security architectures that are able to evolve over time, one needs to take into account the fact that almost all assumption one initially had will become false or moot. This means that adaptability and ability to support changes is crucial.

B. Not Fully Justified

The results in this paper cannot be said to be fully supported by the evidence provided in this paper (or in the referenced papers). They results are neither rigorous nor complete. This is to be expected for such a complex issue. Thus, while the results may be valid and true, they will hardly be complete and not always necessary either. That is, the usual "necessary and sufficient" conditions are not really there. Still, experience and empirical evidence should not be discounted, and we advocate that the lessons learned are taken into account, not as mathematical axioms, but inputs to be considered. Therefore, we recommend that scalable evolving security architectures should be designed with these assumption as background.

C. Pervasiveness, Importance and Dependability

This is important in a world where the internet-of-things (IoT) landslide is about to happen and where the systems will be ever more important.

In the wake of the Snowden revelations, it is also clear that cyber-security is under constant pressure, and while we do not want to over-state the Snowden case per se, it should be clear that the cyber-war methods will (over time) become available to many organizations and individuals. Schneier captures this well when he stated that [71]:

> And technology is fundamentally democratizing: today's NSA secret techniques are tomorrow's PhD theses and the following day's cybercrime attack tools.

How stable and durable are our ICT-based future? The internet pioneer Vinton Cerf warns of a "forgotten century", pointing to the risk that the digital material we produce today may be unreadable by tomorrow equipment [72]. He calls out for "digital vellum" to solve this problem. The risk is no less dire for other reasons for ICT collapse, including Black Swan type of massive security failures.

Finally, it is clear that large-scale ICT infrastructures are highly complex and interdependent entities. The security architectures are no less complex. To name a few, we have issues with backwards compatibility and deprecation of old features, issues with migration towards new functionality, issues with integration with other system, ever-changing threats and ever-changing population of users, etc. In short, it is staggeringly complex, and while little of the complexity is the of the "necessarily complex" type, it is not easily reducible either.

D. Forward Directions

In some sense we find ourselves in the same situation as the old Norse Allfather god Odin. He, with his brothers, had created the world, but it turned out he did not understand his own creation. Odin had to sacrifice one eye to drink from the wisdom well Mimir to gain knowledge.

Our ICT systems are highly complex, relatively fragile and strangely resilient at the same time. We also know that there is a cyber security battlefield and we know we are getting ever more dependent on our systems. So, preferably without too much sacrifice, we urgently need to learn more about what works and what does not work in the protection of our critical ICT infrastructures.

There are obviously technical aspects that needs to be studied further, but this is not enough. We also need a better understanding of the societal aspects of security architecture evolution in large-scale critical ICT system.

IX. Conclusion

In this paper, we have investigated the security architectures or the 3GPP systems. Section II is devoted to this topic. In Section III, we focused on how the security architecture must evolve with the systems and a number of aspects that must be considered. Evolution implies changes, and we have also taken a look at some of the reasons one may want or may have to change the security architecture. This is captured in Section IV. In Section V, we presented a whole range of assumptions about the target systems, the security components and the cryptographic primitives. They are not important individually, but we have postulated them as a means to set up a grand picture of what one must keep in mind regarding a large and long-lived critical infrastructure system.

Somewhat loosely inspired by the article "Why Cryptosystems Fail" [32], we provided a whole section of indirect reasons why things may fail. This is captured in Section VI. The arguments and cases presented here do not constitute evidence or proof. We believed, however, that an awareness of these cases and phenomena may be useful for the mindset needed for designing evolving security architectures. In Section VII, we have tried to distill some of the lessons learned from the 3GPP systems. It is by nature incomplete, but may serve as a starting point for a principles and guidelines document for security architecture design. This section contain discussions, but not all aspects are covered in full. In Section VIII, we include a brief discussion of remaining matters.

Overall, our method has largely been a descriptive one, and a deeper theory of security architecture evolution is still missing. One reason for this is that one in all likelihood cannot fully understand this type of system evolution in terms of security methodologies alone.

The lesson learned, it is hoped, should not be isolated to the 3GPP systems, but be applicable to any system of similar magnitude and scope.

REFERENCES

[1] G. M. Køien, "Challenges for Evolving Large-Scale Security Architectures," in Proceedings of the Eighth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2014), Novemer 16-20, 2014, Lisbon, Portugal. IARIA, 2014, pp. 173–179, ISBN: 978-1-61208-376-6, ISSN: 2162-2116.

[2] 3GPP, TS 43.020, "Security related network functions," 3GPP, France, TS 43.020 (2G), 2014.

[3] 3GPP, TS 21.133, "3G security; Security threats and requirements," 3GPP, France, TS 21.133 (3G), 2001.

[4] 3GPP, TS 33.120, "Security Objectives and Principles," 3GPP, France, TS 33.120 (3G), 2001.

[5] 3GPP, TS 33.210, "3G security; Network Domain Security (NDS); IP network layer security," 3GPP, France, TS 33.210 (NDS/IP), 2012.

[6] 3GPP, TS 33.401, "3GPP System Architecture Evolution (SAE); Security architecture," 3GPP, France, TS 33.401 (3G), 2014.

[7] AVANTSSAR project, "Aslan++ specification and tutorial," FP7-ICT-2007-1, Deliverable 2.3 (update), 01 2008, Available: http://www.avantssar.eu/ [accessed: 2015-06-01].

[8] AVISPA project, Automated Validation of Internet Security Protocols and Applications (AVISPA); AVISPA v1.1 User Manual, 1st ed., AVISPA IST-2001-39252, 06 2006, Available: http://www.avispa-project.org [accessed: 2015-06-01].

[9] N. P. Smart, V. Rijmen, M. Stam, B. Warinschi, and G. Watson, "Study on cryptographic protocols," ENISA, Report TP-06-14-085-EN-N, 11 2014.

[10] European Parliament/European Council, "Directive 2009/136/EC of the European Parliament and of the Council of 25 November 2009 amending Directive 2002/22/EC on universal service and users rights relating to electronic communications networks and services, Directive 2002/58/EC concerning the processing of personal data and the protection of privacy in the electronic communications sector and Regulation (EC) No 2006/2004 on cooperation between national authorities responsible for the enforcement of consumer protection laws." EU, Directive 09/136/EC, 2009.

[11] European Council, "European Council Resolution January 1995 JAI 42 Rev 28197/2/95 (Official Journal reference 96C 329/01 4 November 1996)," EU, Resolution, 1995.

[12] European Parliament/European Council, "Directive 2006/24/EC of the European Parliament and of the Council of 15 March 2006 on the retention of data generated or processed in connection with the provision of publicly available electronic communications services or of public communications networks and amending Directive 2002/58/EC," EU, Directive 06/24/EC, 2006.

[13] J. R. Rao, P. Rohatgi, H. Scherzer, and S. Tinguely, "Partitioning attacks: or how to rapidly clone some gsm cards," in Proceedings of IEEE Symposium on Security and Privacy (2002). IEEE, 2002, pp. 31–41.

[14] M. Kalenderi, D. Pnevmatikatos, I. Papaefstathiou, and C. Manifavas, "Breaking the gsm a5/1 cryptography algorithm with rainbow tables and high-end fpgas," in Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on. IEEE, 2012, pp. 747–753.

[15] P. Papantonakis, D. Pnevmatikatos, I. Papaefstathiou, and C. Manifavas, "Fast, fpga-based rainbow table creation for attacking encrypted mobile communications," in Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on. IEEE, 2013, pp. 1–6.

[16] G. M. Køien, Entity authentication and personal privacy in future cellular systems. River Publishers, 2009, vol. 2.

[17] F. van den Broek, B. Hond, and A. Cedillo Torres, "Security Testing of GSM Implementations," in Engineering Secure Software and Systems, ser. Lecture Notes in Computer Science, J. Jürjens, F. Piessens, and N. Bielova, Eds. Springer International Publishing, 2014, vol. 8364, pp. 179–195.

[18] G. M. Køien, "An introduction to access security in UMTS," Wireless Communications, IEEE, vol. 11, no. 1, Feb 2004, pp. 8–18.

[19] V. Niemi and K. Nyberg, UMTS Security. John Wiley & Sons, 2003.

[20] 3GPP, TS 33.102, "3G Security; Security architecture," 3GPP, France, TS 33.102 (3G), 2014.

[21] 3GPP, TS 33.310, "Network Domain Security (NDS); Authentication Framework (AF)," 3GPP, France, TS 33.310 (NDS/AF), 2014.

[22] G. M. Køien, "Privacy enhanced cellular access security," in Proceedings of the 4th ACM workshop on Wireless security. ACM, 2005, pp. 57–66.

[23] N. Golde, K. Redon, and J.-P. Seifert, "Let me answer that for you: Exploiting broadcast information in cellular networks," in Proceedings of the 22nd USENIX conference on Security. USENIX Association, 2013, pp. 33–48.

[24] N. Gobbo, F. Palmieri, A. Castiglione, M. Migliardi, and A. Merlo, "A denial of service attack to UMTS networks using SIM-less devices," IEEE Transactions on Dependable and Secure Computing, 2014, p. 1.

[25] D. Forsberg, G. Horn, W.-D. Moeller, and V. Niemi, LTE security. John Wiley & Sons, 2012, vol. 1.

[26] A. Gupta, T. Verma, S. Bali, and S. Kaul, "Detecting MS initiated signaling DDoS attacks in 3G/4G wireless networks," in Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on. IEEE, 2013, pp. 1–60.

[27] J. Scahill and J. Begley, "How spies stole the keys to the encryption castle," The Intercept, 2 2015, Available: https://firstlook.org/theintercept/2015/02/19/great-sim-heist/ [accessed: 2015-06-01].

[28] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson, "The matter of heartbleed," in Proceedings of the 2014 Conference on Internet Measurement Conference, ser. IMC '14. New York, NY, USA: ACM, 2014, pp. 475–488.

[29] Y. Huang and A. K. Ghosh, "Introducing diversity and uncertainty to create moving attack surfaces for web services," in Moving Target Defense, ser. Advances in Information Security, S. Jajodia, A. K. Ghosh, V. Swarup, C. Wang, and X. S. Wang, Eds. Springer New York, 2011, vol. 54, pp. 131–151.

[30] M. M. Williamson and J. Léveillé, "An epidemiological model of virus spread and cleanup," Information Infrastructure Laboratory, HP Laboratories Bristol, vol. 27, 2003.

[31] L. Carroll, Through the looking glass: And what Alice found there. Rand, McNally, 1917.

[32] R. Anderson, "Why cryptosystems fail," in Proceedings of the 1st ACM Conference on Computer and Communications Security. ACM, 1993, pp. 215–227.

[33] D. Florêncio and C. Herley, "Where do all the attacks go?" in Economics of Information Security and Privacy III. Springer, 2013, pp. 13–33.

[34] 3GPP, TS 35.202, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification (Release 12)," 3GPP, France, TS 35.202, 2014.

[35] 3GPP, TR 33.908, "3G Security; General report on the design, specification and evaluation of 3GPP standard confidentiality and integrity algorithms," 3GPP, France, TR 33.908, 2001.

[36] 3GPP, TS 35.216, "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2; Document 2: SNOW 3G specification (Release 12)," 3GPP, France, TS 35.216, 2014.

[37] D. Dolev and A. C. Yao, "On the Security of Public-Key Protocols," IEEE Transactions on Information Theory, vol. 29, no. 2, 3 1983, pp. 198–208.

[38] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks," Communications Surveys & Tutorials, IEEE, vol. 15, no. 4, 2013, pp. 2046–2069.

[39] I. Jacobs and N. Walsh, "Architecture of the world wide web, volume one," World Wide Web Consortium (W3C), W3C Recommendation, 12 2004.

[40] R. Anderson, Security engineering. John Wiley & Sons, 2008.

[41] G. M. Køien, "Privacy enhanced mutual authentication in LTE," in Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on. IEEE, 2013, pp. 614–621.

[42] ——, "Mutual entity authentication for LTE," in Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International. IEEE, 2011, pp. 689–694.

[43] European Parliament/European Council, "Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data." EU, Directive 95/46/EC, 1995.

[44] M. Mccauley, The Rise and Fall of the Soviet Union, ser. Longman History of Russia. Taylor & Francis, 2014.

[45] R. Swedberg, "The structure of confidence and the collapse of Lehman Brothers," Research in the Sociology of Organizations, vol. 30, 2010, pp. 71–114.

[46] J. C. Dvorak, "AltaVista, the Biggest Fail Ever," PC Mag, 07 2013.

[47] N. N. Taleb, Fooled by Randomness: The Hidden Role of Chance in Life and in the Markets. Random House Publishing Group, New York, 2001.

[48] ——, The Black Swan: The Impact of the Highly Improbable. Random House Publishing Group, New York, 2007.

[49] F. van den Broek, "Eavesdropping on GSM: state-of-affairs," CoRR, vol. abs/1101.0552, 2011, 5th Benelux Workshop on Information and System Security (WISSec 2010), November 2010.

[50] A. AtKisson, Believing Cassandra: An optimist looks at a pessimist's world. Chelsea Green Publishing Company, 1999.

[51] J. Kruger and D. Dunning, "Unskilled and Unaware of It: How Difficulties in Recognizing One's Own Incompetence Lead to Inflated Self-Assessments." Journal of personality and social psychology, vol. 77, no. 6, 1999, p. 1121.

[52] C. N. Parkinson and O. Lancaster, Parkinson's law: The pursuit of progress. Readers Union [in association with] John Murray, 1959.

[53] P.-H. Kamp, "Why Should I Care What Color the Bikeshed Is?" Frequently Asked Questions for FreeBSD 7. X, 8. X, and 9. X. FreeBSD, 1999, Available: http://bikeshed.com/ [accessed: 2015-06-01].

[54] A. Melnikov, "Rerun: Elliptic curves - preferred curves around 256bit work factor," CFRG mailing list, 2 2015, Available: http://www.ietf.org/mail-archive/web/cfrg/current/msg06331.html [accessed: 2015-06-01].

[55] H. Berghel, "Cyber Chutzpah: The Sony Hack and the Celebration of Hyperbole," IEEE Computer magazine, vol. 48, 02 2015, pp. 77–80.

[56] B. Schneier, "Beyond fear," Copernicus Book, New York, 2003.

[57] ——, "Reconceptualizing Security," in LISA'08: 22nd Large Installation System Administration Conf, 2008.

[58] R. P. Feynman, "Cargo cult science," in Surely You're Joking, Mr. Feynman, 1st ed. W. W. Norton, 1985, Originally a 1974 Caltech commencement address.

[59] G. Hardin, "The Tragedy of the Commons," science, vol. 162, no. 3859, 1968, pp. 1243–1248.

[60] B. Schneier, Liars and outliers: enabling the trust that society needs to thrive. John Wiley & Sons, 2012.

[61] D. Adams, Life, the Universe and Everything: Hitchhiker's Guide 3. Tor UK, 1984, vol. 3.

[62] B. Latané and J. M. Darley, "Bystander Apathy," American Scientist, 1969, pp. 244–268.

[63] J. M. Darley and B. Latane, "Bystander intervention in emergencies: diffusion of responsibility." Journal of personality and social psychology, vol. 8, no. 4p1, 1968, p. 377.

[64] M. Pease, R. Shostak, and L. Lamport, "Reaching Agreement in the Presence of Faults," Journal of the ACM (JACM), vol. 27, no. 2, 1980, pp. 228–234.

[65] L. Lamport, R. Shostak, and M. Pease, "The Byzantine Generals Problem," ACM Transactions on Programming Languages and Systems (TOPLAS), vol. 4, no. 3, 1982, pp. 382–401.

[66] P. Cvitanović, R. Artuso, R. Mainieri, G. Tanner, and G. Vattay, Chaos: Classical and Quantum. Copenhagen: Niels Bohr Institute, 2012, Available: http://chaosbook.org/ [accessed: 2015-06-01].

[67] M. Scheffer, S. R. Carpenter, T. M. Lenton, J. Bascompte, W. Brock, V. Dakos, J. Van De Koppel, I. A. Van De Leemput, S. A. Levin, E. H. Van Nes et al., "Anticipating critical transitions," science, vol. 338, no. 6105, 2012, pp. 344–348.

[68] A. J. Veraart, E. J. Faassen, V. Dakos, E. H. van Nes, M. Lürling, and M. Scheffer, "Recovery rates reflect distance to a tipping point in a living system," Nature, vol. 481, no. 7381, 2012, pp. 357–359.

[69] J. Chew, "Fortune 500 Extinction," csinvesting.org, 01 2012, Available: http://csinvesting.org/2012/01/06/fortune-500-extinction/ [accessed: 2015-06-01].

[70] A. Shostack, Threat modeling: Designing for security. John Wiley & Sons, 2014.

[71] B. Schneier, "Why the NSA's Attacks on the Internet Must Be Made Public," The Guardian.com, 10 2013, Available: http://www.theguardian.com/commentisfree/2013/oct/04/nsa-attacks-internet-bruce-schneier [accessed: 2015-06-01].

[72] I. Sample, "Google boss warns of 'forgotten century' with email and photos at risk," The Guardian.com, 2 2015, Available: http://www.theguardian.com/technology/2015/feb/13/google-boss-warns-forgotten-century-email-photos-vint-cerf [accessed: 2015-06-01].

The Policy-Based AS_PATH Verification to Prevent 1-Hop AS Path Hijacking By Monitoring BGP Live Streams

Je-Kuk Yun, Beomseok Hong, Yanggon Kim
Information Technology
Towson University
Towson, U.S.A.
jyun4, bhong1@students.towson.edu, and ykim@towson.edu

Abstract— **As the number of IP prefix hijacking incidents has increased, many solutions are proposed to prevent IP prefix hijacking, such as RPKI, BGPmon, Argus, and PHAS. Except RPKI, all of the solutions proposed so far can protect ASes only through the origin validation. However, the origin validation cannot detect specified attacks that alter the AS_PATH attribute, such as AS Insertion attack and Invalid AS_PATH Data Insertion attack. In order to solve these problems, the SIDR working group proposed the RPKI using BGPsec, but BGPsec is currently a work in progress. So, we propose Secure AS_PATH BGP (SAPBGP) in which we monitor the AS_PATH attribute in BGP update messages whether each AS in the AS_PATH attribute are connected to each other based on our policy database collected from RIPE NCC repository. Our analysis shows 1.67% of the AS_PATH attributes is invalid and 98.33% of the AS_PATH attributes is valid based on original data including duplication from the ninth of February in 2014 to the fifth of February in 2015. In addition, our results state that 94.41% of the AS_PATH attributes is invalid and 94.41% of the AS_PATH attributes is valid after removing duplicated the AS_PATH attributes. We conducted the performance test and it verified that the SAPBGP can process all of the live BGP messages coming from BGPmon in real time.**

Keywords- border gateway protocol; interdomain routing; network security; networks; AS path hijacking.

I. INTRODUCTION

The Border Gateway Protocol (BGP) is the de-facto protocol to enable large IP networks to form a single Internet [1]. The main objective of BGP is to exchange Network Layer Reachability Information (NLRI) among Autonomous Systems (ASes) so that BGP routers can transfer their traffic to the destination.

However, BGP itself does not have mechanisms to verify if a route is valid because a BGP router completely trusts other BGP routers. This lack of consideration of BGP vulnerabilities often causes severe failures of Internet service provision [3]. The most well-known threat of the failures is the YouTube hijacking by Pakistan Telecom (AS17557) on the 24th of February in 2008 [4]. In response to the government's order to block YouTube access within their ASes, Pakistan Telecom announced a more specific prefix than YouTube prefix. Then, one of Pakistan Telecom's upstream providers, PCCW Global (AS3491), forwarded the announcement to other neighbors. As a result of this, YouTube traffic from all over the world was misled to Pakistan Telecom (AS17557) for two hours. In order to solve these problems, many studies were conducted, such as Resource Public Key Infrastructure (RPKI) [5], BGPmon [6], Argus [7], and a Prefix Hijack Alert System (PHAS) [8].

While there are many studies to IP prefix hijacking, few studies have been researched about AS path hijacking. There was some misdirected network traffic suspected of the man-in-the-middle (MITM) attack in 2013 observed by Renesys. In February 2013, global traffic was redirected to Belarusian ISP GlobalOneBel before its intended destination and it occurred on an almost daily basis. Major financial institutions, governments, and network service providers were affected by this traffic diversion in several countries including the U.S. From the thirty first of July to the nineteenth of August, Icelandic provider Opin Kerfi announced origination routes for 597 IP networks owned by a large VoIP provider in the U.S through Siminn, which is one of the two ISPs that Opin Kerfi has. However, this announcement was never propagated through Fjarskipti which is the other one of the two ISPs. As a result, network traffic was sent to Siminn in London and redirected back to its intended destination. Several different countries in some Icelandic autonomous systems and belonging to the Siminn were affected. However, Opin Kerfi said that the problem was the result of a bug in software and had been resolved [9]. In addition, The Dell SecureWorks Counter Threat Unit (CTU) research team discovered a repeated traffic hijacking to Bitcoin mining sites between February and May 2014. Compromised networks belong to Amazon, Digital Ocean, OVH, etc. The attacker hijacked cryptocurrency miners' traffic and earned an estimated $83,000 [10]. Furthermore, AS 23274, owned by China Telecom, announced approximately 50,000 prefixes, which are registered to other ASes in 2010. The reason the incident was being magnified is because China Telecom is the 11th largest Internet provider. If small ISPs hijacks a large part of the Internet, they do not have the capacity to deal with a huge amount of traffic. China Telecom, however, has the capability to operate under such traffic, and redirect its desired destination. The incident was not recognized for 18 minutes[11]. A root cause of BGP hijacking can be discovered by empirical data analysis using BGP updates from Routeviews, RIB from iPlane project, paths from traceroute, etc. However, proving a malicious intent is hardly possible. According to this research, China Telecom incident is most likely caused by a routing table leak [9].

In order to protect the AS path hijacking, the AS_PATH attribute should not be manipulated. However, the BGP itself cannot check whether the AS_PATH attribute has been changed or not. If a routing hijacker manipulates the AS_PATH attribute in a BGP message that is sent by another router and forwards the manipulated BGP message to other neighbors, the neighbors who receive the manipulated BGP message can be a victim of AS path hijacking. Only Secure Inter-Domain Routing (SIDR) working group proposed the RPKI using BGPsec to validate the AS_PATH attribute. However, BGPsec is currently a work in progress [11]. In addition, a study propounds that BGP armed with BGPsec cannot be secured because of BGP's fundamental design [13].

We proposed Secure AS_PATH BGP (SAPBGP) in which the SAPBGP constructs its own policy-based database by collecting RIPE NCC repository and checks the AS_PATH attribute in BGP update messages whether or not the ASes listed in the AS_PATH attribute are actually connected. We extended the previous study to conduct experiments with increased period of collecting the BGP routing policy data [1]. For the validation test with the real BGP messages, the SAPBGP receives live BGP streams from the BGPmon project [14]. In addition, we conduct the performance test of the SAPBGP to measure the duration of the validation with the live BGP messages.

In this paper, we introduce current active studies on BGP security in Section II. With the fact that BGP is vulnerable to MITM attack, we describe an attack scenario and a solution in Section III. In Section IV, we introduce and explain the SAPBGP in detail. We discuss the SAPBGP environment and analyze the result of the SAPBGP validation and the performance test in Section V. Lastly, we conclude the paper in Section VI.

II. RELATED RESEARCH

A. *Origin validation*

The origin validation is to verify whether the originator of update message has been authorized to announce its prefixes. In order to validate originators, the Resource Public Key Infrastructure (RPKI) is implemented by SIDR working group on January in 2013 and is currently used for origin validation. RPKI is a Public Key Infrastructure (PKI) [15], [16] where an organization called IANA manages officially verifiable Internet resources that are the allocation of hierarchy of IP addresses, Autonomous System Numbers (ASN), and signed objects for routing security. IANA is the trust anchor who allows third party to officially validate assertions according to resource allocations. The authorization is hierarchically assigned from IANA to the Regional Internet Registries (RIRs), Local Internet Registries (LIRs), National Internet Registries (NIRs), and Internet Service Providers (ISPs) as shown in Figure 1.

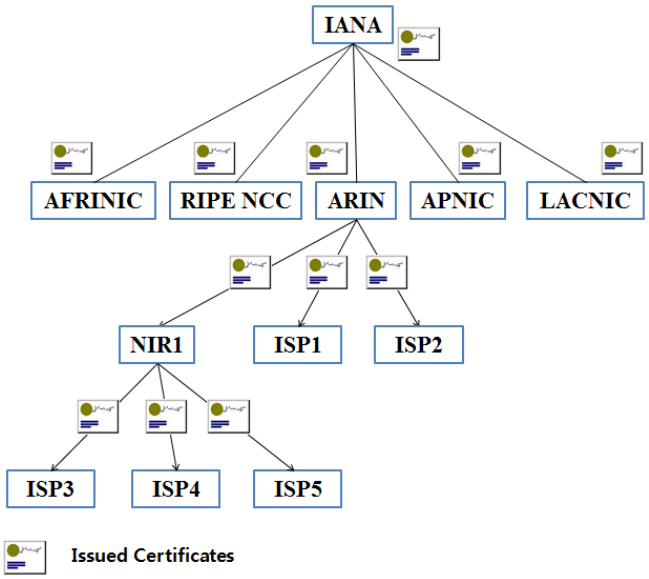


Figure 1. Hierarchy of the RPKI

There are five RIRs and they act as trust anchors like IANA. The RIR issues certificates to NIR, ISP and subscribers. NIR and ISP are allowed to issue certificates to downstream providers and to subscribers. IP address holders specify which ASes are authorized to announce their own IP address prefixes.

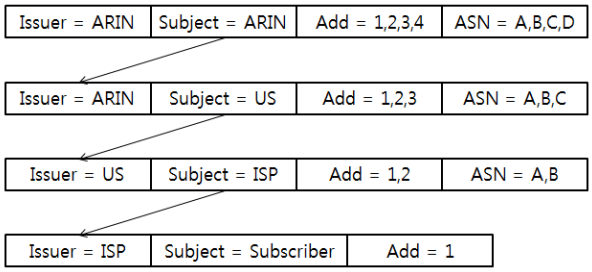


Figure 2. Certificate Chain

Figure 2 explains how a subscriber hierarchically gets certificates regarding their IP address. For example, ARIN issues certificates for US regarding addresses 1, 2, and 3 and ASN A, B, and C as shown in Figure 2. US issues certificates to ISP regarding addresses 1 and 2 and ASN A and B. Then, a subscriber can get a certificate from ISP regarding its IP addresses. As shown in Figure 3, the certificate, called Route Origin Authorizations (ROAs) [17] is a digital object formatted following the Cryptographic Message Syntax Specification (CMS) [18] and composes origin AS Number, validity date range, and one or more IP addresses with a CIDR block. If the address space holder needs to authorize multiple ASes and the IP prefixes are same, the holder should issues multiple ROAs.

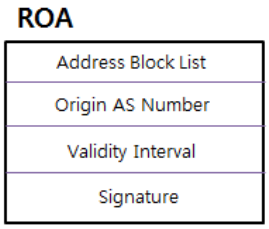


Figure 3. ROA Format

The value of Address Block List is more than one prefixes, corresponding to the NLRI that the ROA signer authorizes for prefix announcements by one or more ISPs. The value of origin AS number that is authorized to announce the prefixes indicated in the address block list. Validity interval indicates the start and end date for which the ROA is valid. Signature includes pairs of information that is used to verify the ROA. One is certificate pointer that directs its parent so that the certificate has been issued by CA. The other one is signature that is digitally signed hash data including address block list, origin as number, validity interval, hash algorithm, and digital signature algorithm. Therefore, if prefix hijacker announce other's prefixes, other network operators can check whether the announcement is invalid after comparing the IP prefixes, ASN included in the update message to the ROA

For example, as shown in Figure 4, there are five ASes. Towson University, AS 6059, announced its prefix 204.62.48.0/22. As the update message is transferred, each ASN is added to the AS_PATH attribute, and finally Verizon receives the update message and knows how to reach the prefix 204.62.48.0/22 through the AS_PATH attribute. However, if a hijacker router, AS 7922, sends the same prefix 204,62,48.0/22, then Verizon will choose AS 7922 as the final destination because the number of hops is shorter than the other as shown in Figure 4.

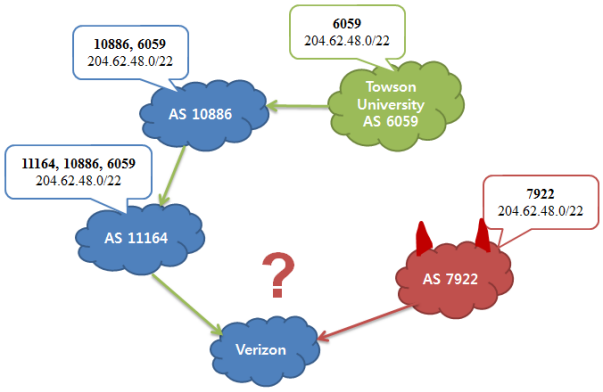


Figure 4. Scenario of IP hijacking

At this moment, if Verizon maintains ROAs and checks the ROAs then Verizon will realize that AS 7922 is not authorized to originate the prefix 204.62.48.0/22 because the ROA as shown in Table I indicates that AS 6059 has been authorized to announce the prefix 204.62.48.0/22. As a result, Verizon can choose the other route as the best path and Internet traffic will be transferred to AS 6059, which is the right destination.

TABLE I. AS 6059'S ROA

ROA
204.62.48.0/22
AS 6059

If every address spaces are authorized by its address holders, then IP prefix hijacking will be fully prevented by RPKI.

B. *BGPsec*

A SIDR working group is designing BGPsec to cryptographically prevent the AS-PATH hijacking [19]. In BGPsec, an optional and non-transitive path attribute, BGPsec_Path attribute, is included in BGP update messages. BGPsec depends on RPKI certificates and BGP router, which wants to send a BGP update messages that including the BGPsec_Path should have a private key associated with the BGP router's AS number. When the BGP router originates IP prefixes, the BGP router signs the update message with its private key so that any BGP router that receives the update message can check that the update message has been originated by the right BGP router by verifying the signature with the public key corresponding to the private key. In addition, BGP routers that receive the BGP update message sign the BGP update message with their private key and forward the BGP update message to neighbors. If every router that receives and forwards the BGP update messages signs the BGP update message, the BGP update message can be considered as the message that has not been synthesized by hijackers.

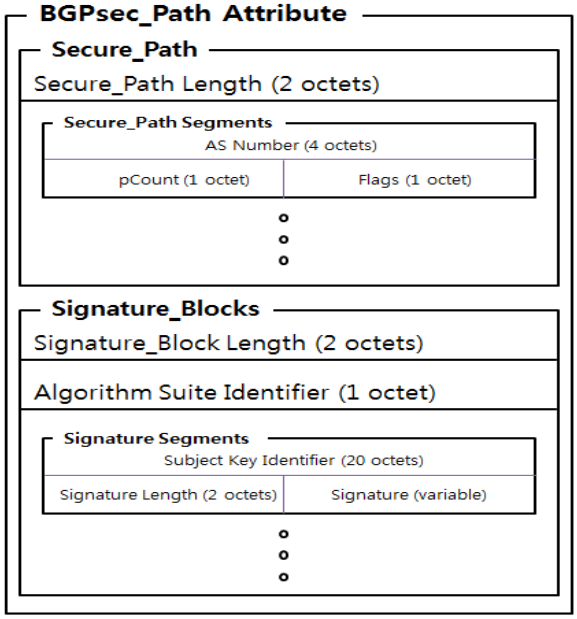


Figure 5. BGPsec_Path Attribute

In order to protect BGP update message, especially to protect AS_PATH attributes, the BGP update message should carry the secured information such as digital signature.

We call the BGP update message including a BGPsec_Path attribute BGPsec update messages as shown in Figure 5. The AS_PATH attribute in BGP update messages is replaced with BGPsec_Path attribute in the BGPsec update messages

BGPsec relies on RPKI where the root of trust consists of the Regional Internet Registries (RIRs), including ARIN, LACNIC, APNIC, RIPE, and AFRINIC. Each of the RIRs signs certificates to allocate their resources. RPKI provides ROA to ASes that are authorized to advertise a specific prefix. The ROA contains the prefix address, maxlength, and AS number, which certifies the specified AS has permission to announce the prefixes. For routing path validation, each AS receives a pair of keys, which are a private key and a public key, from its RIR. Each AS speaker signs the routing path before forwarding it to their neighbors.

C. BGPmon

BGPmon is a monitoring infrastructure, implemented by Colorado State University that collects BGP messages from various routers that are distributed and offers the BGP messages as the routes for destinations are changed in real-time [14]. Any BGP router can be a source that offers real-time update messages if the BGP router is connected to BGPmon. Currently, 9 organizations participate in the BGPmon project as a source router. In addition, BGPmon collects Multi-threaded Routing Toolkit (MRT) format [20] live streams from the RouteViews project through indirect peering. The MRT format defines a way to exchange and export routing information through which researchers can be provided BGP messages from any routers to analyze routing information. Clients can be connected to the BGPmon via telnet and receive the live BGP stream in real time.

D. RIPE NCC

RIPE NCC is one of the Regional Internet Registries (RIRs) in charge of the Europe/Middle-East region. RIPE NCC manages RIPE Data Repository that is a collection of datasets, such as IP address space allocations and assignments, routing policies, reverse delegations, and contacts for scientific Internet research. The original purpose of the BGP policy is to filter incoming BGP messages and to choose BGP peers that will receive the BGP messages using BGP import and export policies. BGP router operators voluntarily upload their BGP policies to Internet Route Registries (IRR) through a predefined format, called Routing Policy Specification Language (RPSL) [20] that is provided by IRR. RIPE NCC database has been part of IRR and is composed of a set of online databases that is available for research purposes. In addition, RIPE NCC monitors Internet routing data and stores links between the routing data that has been seen by RIPE NCC. RIPE NCC provides users with RIPE Data Repository that contains BGP peer information. Through this information, we can know if any ASes are connected to other ASes. This peer information has been collected by either Routing Information Service (RIS) or IRR. RIS has collected and stored Internet routing data from several locations all over the world since 2001. The organizations or individuals who currently hold Internet resources are responsible for updating information in the database.

III. BGP Threats and Solution

In this section, we introduce a scenario of the AS path hijacking that leads to the MITM attack. In addition, we discuss how the routing policy-based AS_PATH validation is operated in order to prevent the AS path hijacking.

A. Manipulating data in BGP updates

A BGP router inserts its own ASN into the AS_PATH attribute in update messages when the BGP router receives the update message from neighbors. However, the BGP router can insert one or more ASNs into the AS_PATH attribute in update messages other than its own ASN. In addition, a BGP router might pretend as if the BGP router is connected to a certain BGP router by manipulating data contained in BGP updates.

Figure 6 demonstrates an example of manipulating data in BGP update messages. Suppose AS 400 has a connection to AS 500 and creates a fake BGP announcement to pretend that AS 400 received a BGP message originated by AS 100 and forwarded the update message to AS 500 even though AS 100 and AS 400 actually do not have a BGP connection.

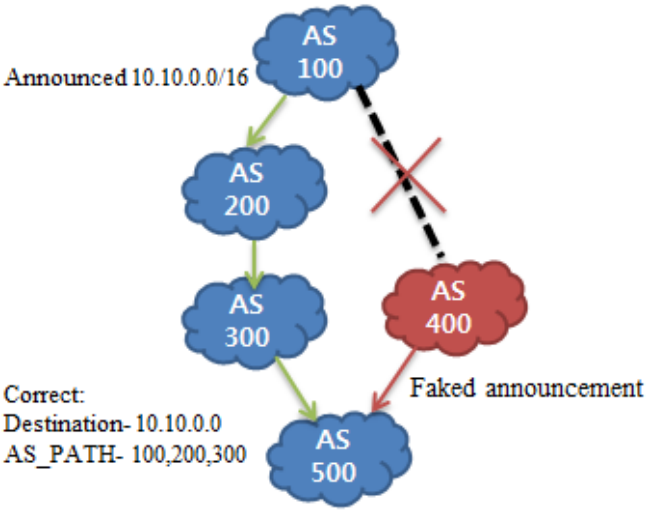


Figure 6. Manipulating a BGP message

In terms of AS 500, the traffic heading for prefix 10.10.0.0/16 will choose AS 400 as the best path because AS 500 selects the shortest path and AS 400 is shorter than AS 300. Even if the AS 500 can conduct origin validation, the AS 500 cannot prevent this attack because prefix and ASN information is correct. As a result, AS 400 will have the traffic heading for prefix 10.10.0.0 and might start another attack using the traffic, such as a MITM attack.

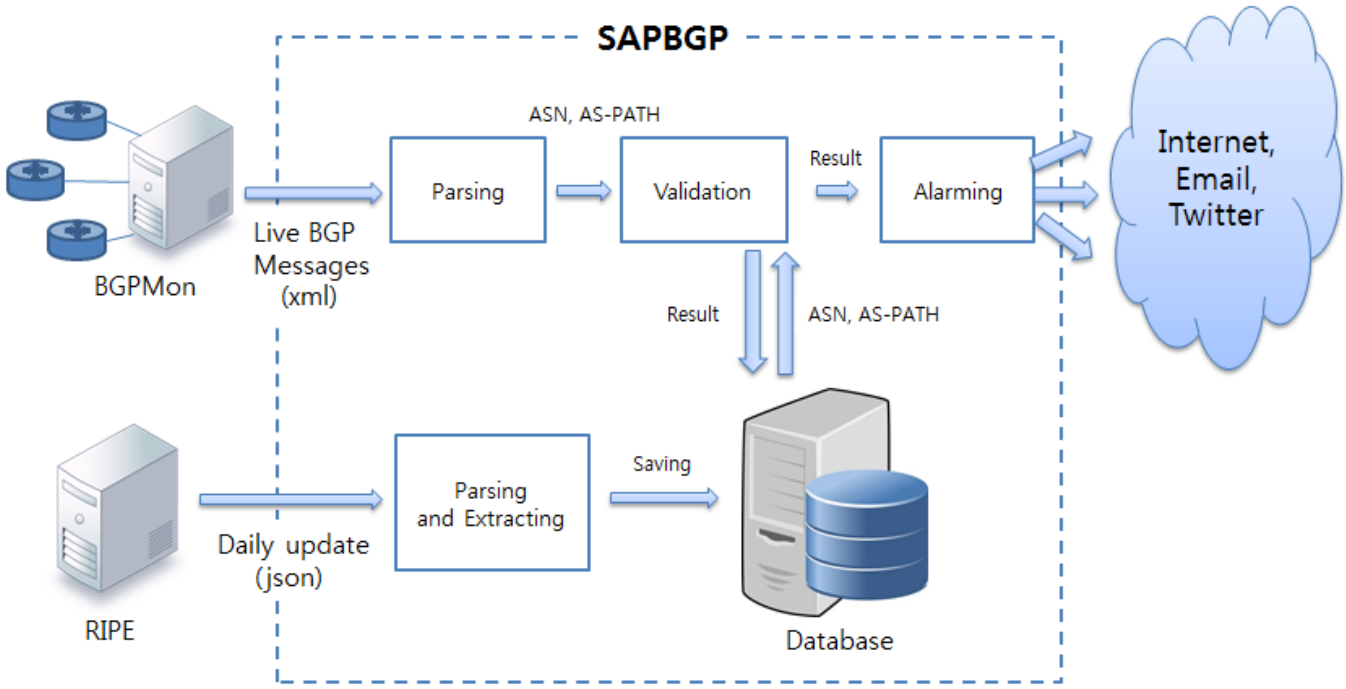


Figure 7. The architecture of the SAPBGP

B. *Man-in-the-middle (MITM) attack*

The man-in-the-middle attack is an active eavesdropping in which the attacker secretly creates connections to the victims and redirects large blocks of internet traffic between the sources and the destinations as if the sources and destinations communicate directly. In such a case, the victims can only notice a little enlarged latency time because the internet packets travel longer hops than normal. In the meantime, the attacker can monitor and manipulate the packets so that the attacker can create future chances to try another attack.

Renesys monitors the entire internet and they inform the targeted networks of hijacking incidents. With the support from LINX(London Internet Exchange) and other IXPs(Internet Exchange Point), they can make a more correct judgment over the hijacking. Renesys found MITM attacks and its clients were victims of route hijacking caused by MITM attacks for more than 60 days. The victims are governments, Internet Service Providers (ISPs), financial institutions, etc. [9]. Renesys detected several AS path hijacking attempts: Beltelecom (AS 6697) and Siminn (AS 6677) [9] . Victims whose traffic was diverted varied by day, and included major financial institutions, governments, and network service providers. Affected countries included the US, South Korea, Germany, the Czech Republic, Lithuania, Libya, and Iran.

C. *Routing policy based AS_PATH Validation*

RIPE NCC provides users with RIPE Data Repository that contains BGP peer information. Through this information, we can know if any ASes are connected to other ASes. This peer information has been collected by either Routing Information Service (RIS) or Internet Routing Registry (IRR). RIS has collected and stored Internet routing data from several locations all over the world since 2001.

Using peer information, the SAPBGP monitors live BGP streams from BGPmon. For example, in Figure 6, suppose that AS 400 pretends as if AS 400 is connected to AS 100, and AS 400 creates a BGP message as if the BGP message is coming from AS 100 and forwarding the BGP message. Then, AS 500 cannot check AS 400 and AS 100 are connected to each other even though the AS 500 can conduct the origin validation. However, suppose that either AS 500 or one of AS 500's neighbors is a BGPmon's participant and the SAPBGP can receive the live BGP stream related to AS 500. The AS_PATH attribute in the BGP stream should contain AS_PATH-100, 400, 500. Then, the SAPBGP can find that AS 100 and AS 400 are not connected to each other according to the peer information collected from RIPE NCC repository. As a result of this, an AS 500 administrator will be alerted by the SAPBGP and realize AS 400 might be trying the MITM attack to draw AS 500 traffic heading to AS 100.

IV. SECURE AS_PATH BGP

In this section, we introduce overall how the SAPBGP works and Figure 7 describes the architecture of the SAPBGP.

A. *Constructing Database*

We construct our own database by using API provided by RIPE. We have collected, every day, all of the AS imports and exports policies information since the eighteenth of February in 2014. In addition, we have separated tables in the database to keep the daily information as well as the

accumulated information by adding new exports and imports to the existing exports and imports.

When the BGP was designed for the first time, the initial number of bits for the AS number was 16 bits, so AS number ranged from 0 to 65535. However, the number of bits for the AS number was changed to 32 bits. After that, each RIR reserves AS numbers as indicated Table II. We collected policy information from AS 1 to AS 394239 and skipped unallocated AS numbers that are not indicated in Table II.

TABLE II. 32 BITS AS NUMBER ALLOCATION ABOVE 65535

	Allocation	***The number of ASes***
APNIC	131,072-135,580	4,509
RIPE NCC	196,608-202,239	5,632
LACNIC	262,144-265,628	3,485
AFRINIC	327,680-328,703	1,024
ARIN	393,216-394,239	1,024

We sent queries to RIPE NCC one by one. For example, if a query is related to AS 1 then the result includes AS 1's export policies, imports polices, and prefixes in the form of JSON. The SAPBGP parses the results so that the list of export policies and import policies can be stored to AS 1's record in the table. As a result, a new table is created every day to keep track of the daily policy information. In addition, the accumulated table is updated by adding new policies if AS 1 adds new policies against other ASes. Figure 8 shows the records from AS 10001 to AS 10005 in the policy table.

asn	export	import
10001	4680,2497,2516	
10002	2497,17224,9002,4716,251...	17225,4716,17232,45686,4732,10015
10003	4716,6939,2516,2497	4716,2516
10004	7682,4675,4732,4686,2519	7682,4732
10005		

Figure 8. A screen capture of the policy table

B. Monitoring Live BGP Stream

BGPmon provides live BGP streams through telnet to the public. So, whenever the routers that are connected to BGPmon receives BGP update messages, BGPmon converts BGP update messages to XML format messages and propagates the XML format messages to their clients. Apart from the BGP update message, the XML format message includes timestamp, date time, BGPmon id, BGPmon sequence number, and so on.

Currently, there are 9 participants that are directly connected to BGPmon as shown in Table III.

TABLE III. 9 ORGANIZATIONS THAT PARTICIPANTE IN THE BGPMON PROJECT

AS number	***Organization name***
812	Rogers Cable Communication Inc.
3303	Swisscom (Switzerland) Ltd
3257	Tinet SpA (RIPE NCC)
5568	ROSNIIROS Russian Institute for Public Networks
6447	University of Oregon
10876	MAOZ.com
14041	University Corporation for Atmospheric Research
12145	Colorado State University
28289	Americana Digital Ltda.

We measured the number of update messages that BGPmon propagates for 1 hour on the twenty sixth of February in 2014. Table III shows the minimum, maximum, and average number of update messages per 10 seconds.

TABLE IV. THE NUMBER OF UPDATE MESSAGES FROM BGPMON

	The number of update messages per 10 seconds
Minimum	38
Maximum	1,672
Average	119.43

After parsing the live BGP message, the SAPBGP retrieves the ASN attribute and the AS_PATH attribute to check whether ASes in the AS_PATH attribute are connected to each other. Firstly, we compare the policy table in the database that is collected one day before. If we cannot find the pair, we compare the information from the accumulated table. If we cannot find the pair from the table, we consider the AS_PATH attribute as the suspicious AS_PATH attribute. If we find the suspicious AS_PATH attribute, we notify the AS network administrators of the suspicious AS_PATH attribute.

V. PERFORMACE TEST AND RESULT ANALYSIS

We explain the environment in which the SAPBGP constructs its own database by collecting RIPE repository and check the live BGP stream from BGPmon to check the invalid AS_PATH attribute in the BGP message. In addition, we conduct the performance test and analyze the result of the performance test in this section.

A. Experiment

In order to monitor AS path hijacking in the real world, we collected BGP live stream from the BGPmon project and compared the AS_PATH attribute to our policy-based database. The policy-based database is updated daily because BGP policy information changed whenever network operators wanted to change their BGP policies. A new BGP policy table is created every day, so we used the BGP policy table that is collected one day before the day we conducted the experiment. The number of BGP routing policies that are registered by AS holders is 55,395 on February in 2015, which means only 68% of AS holders registered their BGP routing policies as shown in Figure 9.

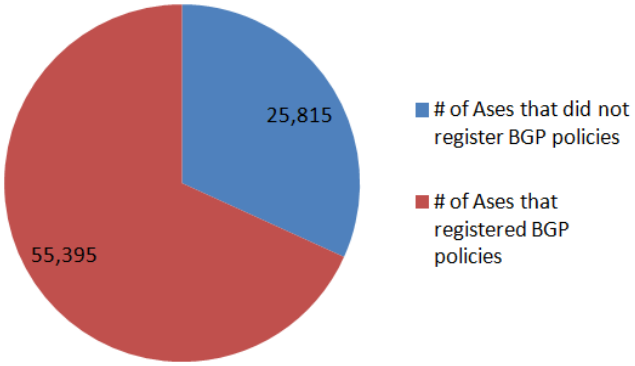


Figure 9. Ratio of ASes that registered BGP routing policies

We have constructed our database by daily collecting BGP policy records from the RIPE repository since the eighteenth of February in 2014. Based on our table, the SAPBGP checked the live BGP streams from BGPmon.

TABLE V. THE COMPARISON OF THE RESULTS

	Duplication included	*No duplication*
Valid	1,950,904	83,636
Invalid	34,938	5,271
Valid by the accumulated records	107,795	5,463

Table V shows the comparison between the original results and the result that does not contain duplications. Because of the difference of variation of BGP update periodic time, some pairs of ASes can be duplicated more than others.

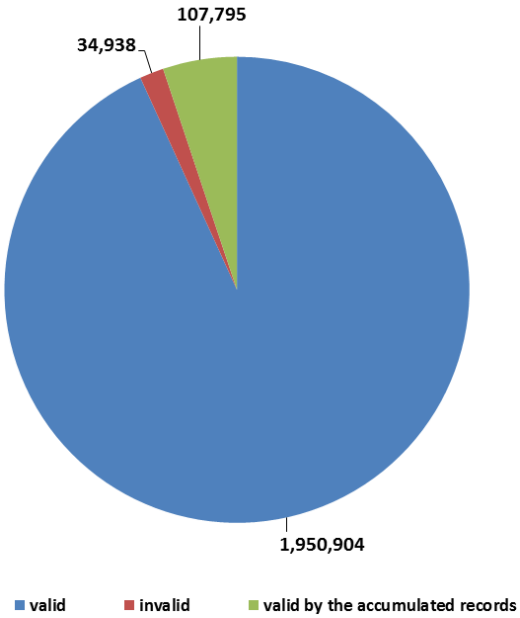


Figure 10. The result of the AS_PATH monitoring experiment that includes duplications

Figure 10 shows the result of the AS_PATH monitoring experiment through the SAPBGP from the ninth of February in 2014 to the fifth of February in 2015. We conducted the experiment randomly twice a month during that period. Figure 10 shows the original data that contains many duplicated results. Our result indicates 1.67% of the AS_PATH attributes are invalid and 98.33% of the AS_PATH attributes is valid.

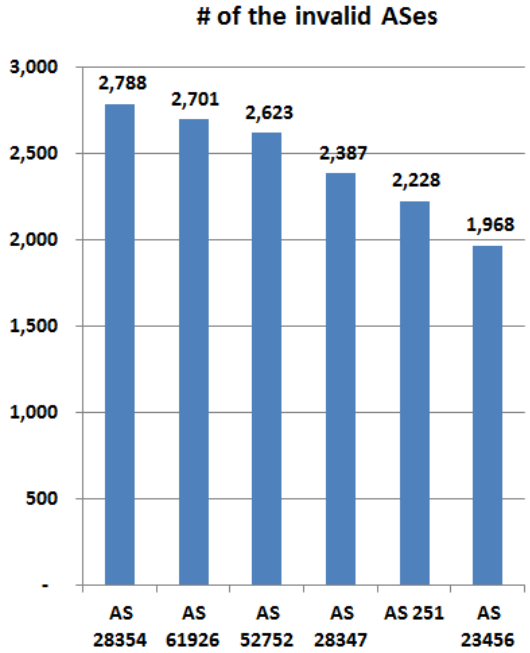


Figure 11. A portion of the policy table of the invalid ASes that includes duplications

Figure 11 illustrates a portion of the policy table of the invalid ASes that the SAPBGP detected in the experiment and this result contains duplications. The invalid ASes could signify either the AS holder does not configure policies or the AS_PATH attribute was manipulated by hijackers.

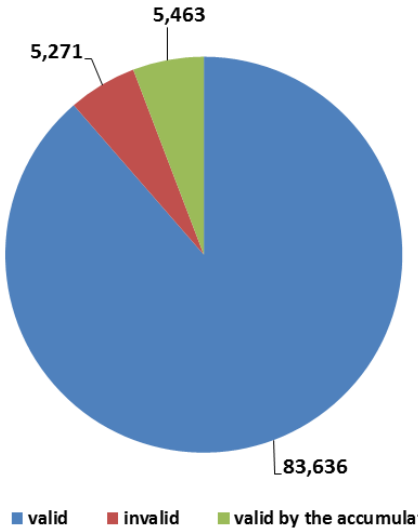


Figure 12. The result of the AS_PATH monitoring experiment that includes duplications

Since original data contains many duplicated information, we analyzed the result that does not contain duplications as well. Figure 12 shows the result of AS_PATH that does not contain the duplications. Our result shows 5.57% of the AS_PATH attribute are invalid and 95.43% of the AS_PATH attribute are valid.

Figure 13 illustrates a portion of the policy table of the invalid ASes that the SAPBGP detected in the experiment. The result does not contain duplications from the original results.

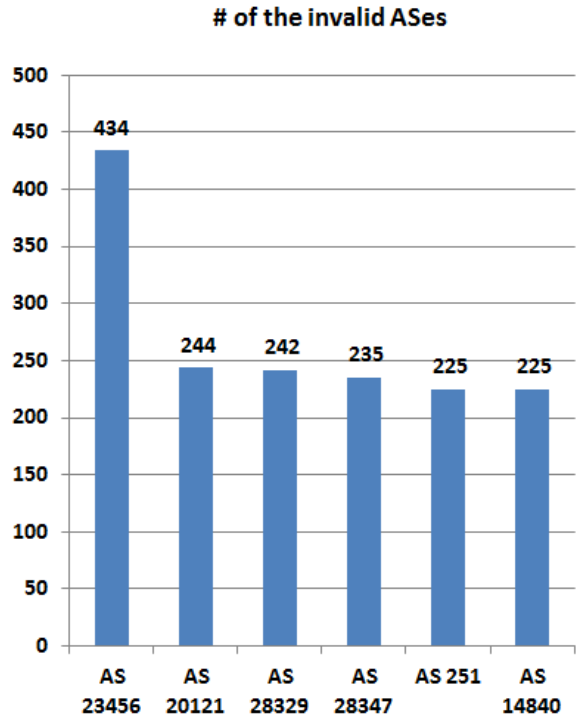


Figure 13. # of ASes that registered BGP policies that do not include duplications

The number of ASes that registered BGP routing policies are gradually increased according to our policy database. The total number of ASes is 81,210 and it will take a long time for every AS holder to register BGP policies. Figure 14 shows how many of ASes that registered BGP policies is increased for 1 year between March in 2014 and February in 2015. In order to check connections between two peers, BGP policy information from each BGP peer should contain the BGP policy against the other peer. However, we considered a BGP connection is valid if only one of two BGP peers has the BGP policy against the other peer because the number of ASes that registered BGP policy is still small. In addition, we considered a BGP message as valid message if one of an AS_PATH pair is the one of 9 organizations that participate in the BGPmon project.

We assumed that a pair of AS_PATH that is invalid and is placed at the second position in the AS_PATH attribute can be candidates of 1-hop hijacker because the number of hops should be shorter than others to draw Internet traffic to their AS. Since the first position is the destination AS, the second position AS can hijack Internet traffic heading for the first position AS.

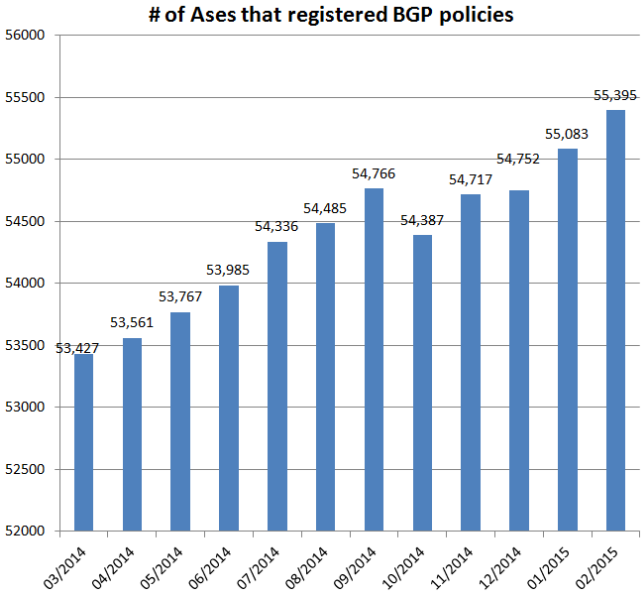


Figure 14. # of ASes that registered BGP policies

Table VI enumerates the top 20 1-hop hijacking candidates.

TABLE VI. TOP 20 1-HOP HIJACKING CANDIDATES

First position	*Second position*	*Frequency*
AS 4739	AS 3491	12
AS 4739	AS 1239	12
AS 4739	AS 1273	12
AS 4739	AS 1299	12
AS 3491	AS 7575	12
AS 4739	AS 209	12
AS 10026	AS 3491	11
AS 10026	AS 1273	11
AS 4739	AS 24115	11
AS 4739	AS 9488	11
AS 53237	AS 12956	11
AS 7575	AS 24490	11
AS 4739	AS 2914	11
AS 4826	AS 2828	11
AS 4739	AS 4635	10
AS 38809	AS 2914	10
AS 4826	AS 9498	10
AS 4739	AS 10026	10
AS 10026	AS 1299	10
AS 53237	AS 3549	10

We checked 94,370 invalid pairs of AS_PATH attributes that do not include duplications and we considered 1-hop hijacking candidates if the pair is located at first and second positions in the AS_PATH attribute.

B. Performance Test

The SAPBGP runs on a 3.40 GHz i5-3570 machine with 16 GB of memory running Windows 7. MySQL Ver. 14.14 Distrib 5.1.41 is used for the database. The SAPBGP is implemented by JAVA to collect daily updates from RIPE, to receive live BGP streams from BGPmon, and to validate the BGP stream by comparing the AS_PATH attribute to our database. The SAPBGP and database are located in the same machine to reduce the connection latency between them.

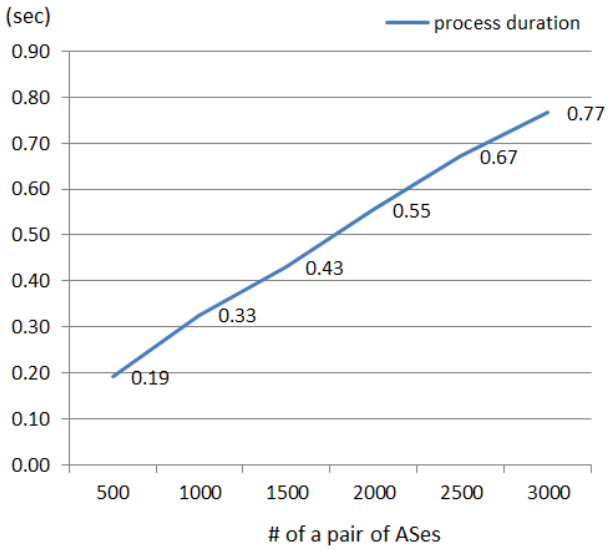


Figure 15. The result of the performance test for the AS_PATH validation

Figure 15 shows the AS_PATH validation time. The validation time includes accessing database, retrieving the specific AS record from a table, and comparing the AS_PATH attribute to the AS's record. We conducted performance test for around 1,864,567 live BGP streams. As shown in Table VI, it takes 4.12 ms, on average, to validate a pair of ASes.

TABLE VII. AS_PATH VALIDATION TIME TO PROCESS ONE BGP UPDATE MESSAGE

	Duration for verifying a BGP message
Minimum	0.07ms
Maximum	9.86sec
Average	4.12ms

According to Table IV, the maximum number of live BGP messages for 10 seconds is 1,672. The SAPBGP can process 2,427.18 BGP messages for 10 seconds, on the average, based on the performance test as shown in Table VII. So, the SAPBGP can process all of the live BGP messages coming from BGPmon in real time.

VI. CONCLUSION

Even though many solutions are proposed to prevent IP prefix hijacking, such as RPKI, BGPmon, Argus, and PHAS, these solutions cannot protect the AS path hijacking except RPKI. SIDR proposed the RPKI using BGPsec, but BGPsec is currently a work in progress. In order to monitor the AS path hijacking, we propose Secure AS_PATH BGP (SAPBGP) in which we monitor the AS_PATH attribute in update messages whether each AS in the AS_PATH attribute are connected to each other based on our policy database collected from RIPE NCC repository. Our analysis shows 1.67% of the AS_PATH attributes is invalid and 98.33% of the AS_PATH attributes is valid based on original data including duplication from the ninth of February in 2014 to the fifth of February in 2015. In addition, our results state that 94.41% of the AS_PATH attributes is invalid and 94.41% of the AS_PATH attributes is valid after removing duplicated the AS_PATH attributes. In addition, the result of the performance test verifies that the SAPBGP can process all of the live BGP messages coming from BGPmon in real time. In the result of the AS_PATH monitoring experiment, the ratio of invalid AS_PATH attribute is high because some AS routers still do not configure their policies. For the precise result of the policy based AS_PATH validation, every router needs to configure policies against its peers.

REFERENCES

[1] J. Yun, B. Hong, and Y. Kim, "The Policy-Based AS_PATH Verification to Monitor AS Path Hijacking," The Eighth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2014), Lisbon, Portugal, 16-20, November, 2014, pp.20-24.

[2] Y. Rekhter, "A Border Gateway Protocol 4 (BGP-4)," 2006, RFC 4271.

[3] S. Murphy, "BGP Security Vulnerabilities Analysis," 2006, RFC 4272.

[4] Rensys Blog, Pakistan hijacks YouTube [Online]. Available: http://www.renesys.com/blog/2008/02/pakistan_hijacks_youtube_1.shtml [Accessed February 2014].

[5] T. Manderson, L. Vegoda, and S. Kent, "Resource Public Key Infrastructure (RPKI) Objects Issued by IANA(Feb. 2012)," 2012, [Online]. Available: http://www.rfc-editor.org/rfc/rfc6491.txt [Accessed January 2014].

[6] BGPmon, Google's services redirected to Romania and Austria [Online]. Available: http://www.bgpmon.net/googles-services-redirected-to-romania-and-austria [Accessed October 2013].

[7] X. Shi, Y. Xiang, Z. Wang, X. Yin, and J. Wu., "Detecting Prefix Hijackings in the Internet with Argus," In Proc. of ACM IMC 2012.

[8] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang, "PHAS: A prefix hijack alert system," 2006, In Proceedings of the 15th conference on USENIX Security Symposium - Volume 15 (USENIX-SS'06), Vol. 15, pp.153-166.

[9] Renesys Blog, Targeted Internet Traffic Misdirection [Online]. Available: http://www.renesys.com/2013/11/mitm-internet-hijacking [Accessed January 2014].

[10] P. Litke and J. Steward,"BGP Hijacking for Cryptocurrency Profit" [Online]. Available: http://www.secureworks.com/cyber-threat-intelligence/threats/bgp-hijacking-for-cryptocurrency-profit [Accessed February 2015]

[11] J. Cowie. Renesys blog: China's 18-minute mystery [Online]. Available: http://www.renesys.com/blog/2010/11/chinas-18-minute-mystery.shtml[Accessed January 2015].

[12] M. Lepinski, Ed., and BBN, "BGPSEC Protocol Specification," Available: http://tools.ietf.org/html/draft-ietf-sidr-bgpsec-protocol-11, [Accessed January 2015].

[13] Q. Li, Y. Hu, and X. Zhang, "Even Rockets Cannot Make Pigs Fly Sustainably: Can BGP be Secured with BGPsec?," 2014.

[14] The BGPmon project, http://bgpmon.netsec.colostate.edu, [Accessed 6th July 2013].

[15] R. Housley, W. Ford, W. Polk, and D. Solo, Internet X.509 Public Key Infrastructure Certificate and CRL Profile, RFC2459, 1999.

[16] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, Internet X.509 Public Key Infrastructure Certificate, and Certificate Revocation List (CRL) Profile, RFC5280, 2008 .

[17] M. Lepinski, S. Kent, and D. Kong, "A Profile for Route Origin Authorizations (ROAs)," [Online]. Available: http://tools.ietf.org/html/rfc6482, [Accessed December 2012].

[18] R. Housley, "Cryptographic Message Syntax (CMS)," [Online]. Available: http://www.ietf.org/rfc/rfc3852.txt, [Accessed September 2014].

[19] IETF. "Secure Inter-Domain Routing (SIDR)". Online, Sep. 2010. Available from http://datatracker.ietf.org/wg/sidr/

[20] L. Blunk, "Multi-Threaded Routing Toolkit (MRT) Routing Information Export Format," RFC 6396 , 2011.

[21] C. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg, and M. Terpstra, "Routing Policy Specification Language (RPSL)," RFC 2622, June 1999.

Using Managed Certificate Whitelisting as a Basis for Internet of Things Security in Industrial Automation Applications

Rainer Falk and Steffen Fries
Siemens AG
Corporate Technology
Munich, Germany
Email: {rainer.falk|steffen.fries}@siemens.com

***Abstract*—Device authentication is a basic security feature for automation systems, and for the future Internet of Things. The design, setup, and operation of a practically usable security infrastructure for the management of required device credentials – as cryptographic device keys, and device certificates – is a huge challenge. Also, access permissions defining authorized communication peers have to be configured on devices. The setup, and operation of a public key infrastructure PKI with registration authority (RA), and certification authority (CA), as well as the management of device permissions has shown to be burdensome for industrial application domains. A recent approach is based on certificate whitelisting. It is currently standardized for field device communication within energy automation systems by IEC 62351 in alignment with ITU-T X.509. This new approach changes the way how digital certificates are used, and managed significantly. After describing the new approach of managed certificate whitelisting, and giving a summary of ongoing standardization activities, an example for the application in a real-world application domain is described. Needs for further technical work are derived, and solution options are presented.**

***Keywords*—Digital certificate; certificate whitelisting; credential management; PKI; device authentication; Internet of Things.**

I. Introduction

Industrial automation control systems (IACS) monitor, and control automation systems in different automation domains, e. g., energy automation, railway automation, or process automation. The main functionality can be summarized on a high level to performing control operations in the physical world using actuators, based on physical measurements obtained by sensors. Automation control systems are using open communication protocols like Ethernet, IP, TCP/UDP internally, and for communication with external systems (e. g., for monitoring, diagnosis, configuration), realizing an Internet of Things (IoT), or the Web of systems. The term "Internet of Things" commonly refers to a set of technologies supporting the connection of hitherto stand-alone devices to an IP-based network. These technologies are important enablers for the convergence of today's automation architectures with service-oriented approaches while meeting industry-grade safety, security, reliability, and real-time requirements.

In a common realization of a public key infrastructure PKI, digital certificates are issued by a trusted certification authority (CA). This allows to authenticate devices. Additionally, access permissions are defined for authorized communication peers. While this technology could be the basis for a global, uniform secure communication, in reality, the deployment, and adoption of PKIs is often limited to HTTP server authentication. A reason for that is the significant effort required to setup, maintain, and use a PKI. Also, differing interests of involved stakeholders prevent that a single security infrastructure can be setup that is relied upon by all stakeholders. The problem addressed in this paper is the practical management of device certificates for field-level automation devices, being an extended version of [1].

Industrial automation control systems use open communication protocols as Ethernet, wireless local area network (WLAN) IEEE 802.11 [2], transmission control protocol (TCP), user datagram protocol (UDP), and hypertext transfer protocol (HTTP) [3]. The communication can be protected using standard security protocols like IEEE 802.1X/MACsec [4], Internet key exchange (IKE) [5] with Internet protocol security (IPsec) [6], secure shell (ssh) [7], secure sockets layer (SSL) [8], and transport layer security (TLS) [9]. Often, asymmetric cryptographic keys, and corresponding device certificates are used. Symmetric keys would not scale well for the huge number of involved devices as pairwise shared secrets would need to be managed. This is be feasible only for a small number of devices.

A certificate infrastructure is required that is suitable for an operational automation environment. Main considerations are the demand for extremely high system availability, requiring that the automation system can continue to operate in an autonomous island mode, and the fact that many automation systems are setup as separate network segments that have no, or only limited connectivity with general office networks, or even the public Internet. Moreover, the fact that these systems are typically engineered, e.g., that the communication relations are known up front, can be leveraged for certificate and access management. It should also be mentioned that certification of products and solutions plays an increasingly important role. Especially in the area of critical infrastructures, regulatory requirements for product certification exist. But also operators require certified products to ensure both their own compliance with defined security procedures, and policies, as well as to ensure product interoperability, and security. The industrial security standard being investigated in this paper is ISO/IEC 62433 [10], which on one hand defines security levels, and on the other hand defines requirements

for each security level, without being prescriptive about the actual implementation. This standard is currently intended to enhance the industrial automation certification scheme of IEC IECEE [11] with respect to security.

Existing approaches for device certificate management have severe limitations when applied for field-level automation devices. A self-contained certificate management tool (command line tool, or with GUI) can be, with corresponding procedures, and personal, and organizational security measures, well suited for a small number of devices, but it does not scale well to scenarios with a larger number of devices. A full-blown PKI infrastructure could be efficient for an extremely huge number of devices, as, e. g., WiMax, or cellular modems, but these go beyond the scale of a common single automation systems.

The problem can be summarized that a solution is needed that can be setup, and operated autonomously within a certain automation environment without relying on a globally accepted certification authority, and that scales well for "mid-size" automation environments, for which a self-contained certificate tool is too small, and a full PKI solution would be too complex, and costly. It may be also advantageous to avoid the need for deploying a separate identity, and access management infrastructure. The paper is an extended version of [1] that presents in particular more details about security for industrial automation, and control systems, and describes extended example for the usage of certificate whitelisting within the energy automation application domain.

The remainder of this paper if structured as follows: After summarizing background work in Section II, an overview on the industrial security standards IEC62443 [10] is given in section III. Section IV describes certificate whitelists as a new paradigm for using digital certificates. The management of certificate whitelists is described generically in Section V, and a specific adaption into energy automation systems is outlined in Section VI. An outlook to possible future extensions is given in Section VII.

II. Background and Previous Work

Secure communication protocols, digital certificates, and public key infrastructure PKI [12], [13] have been dealt with intensively for years. An introduction is given in common text books on IT security [14]. The remainder of this section summarizes shortly major aspects that are relevant to managed certificate whitelists.

A. Device Communication Security Technologies

Digital device certificates are the basis for device communication security as used in industrial automation systems, and in the future Internet of Things (IoT). Major communication security protocols are available for the different layers of the communication protocol stack that support digital device certificates for authentication:

- Link layer: The standard 802.1X [4] provides Network Access Control to restrict access to a network only for authenticated devices. It is also possible to encrypt the communication link using the MACsec of 802.1X.

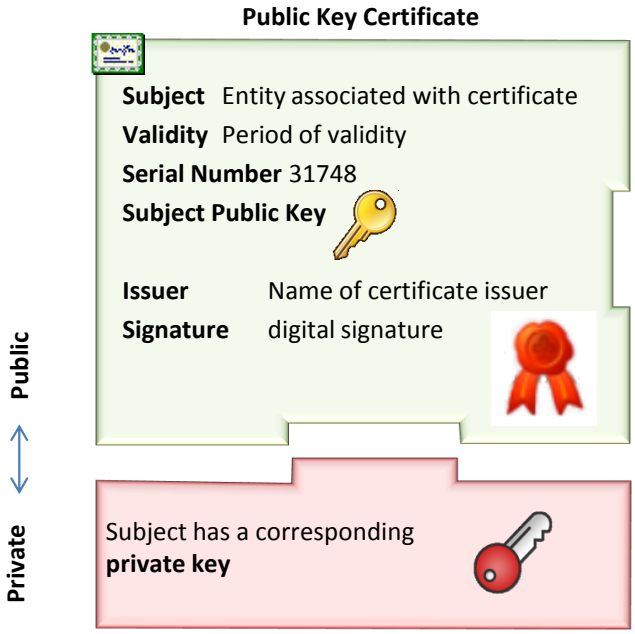


Fig. 1. Digital Certificate (X.509)

- Network layer: The communication can be protected with IPsec [6] on the network layer. The required security associations can be established by the IKE [5] protocol.
- Transport layer: With TLS [9], the successor of the SSL protocol [8], communication can be protected on the transport layer.
- Application layer: SSH, or WS-Sec are available to protect application layer protocols as HTTP, SOA (REST, SOAP), CoAP, XMPP, or MQTT.

B. Digital Certificates

The main purpose of a digital certificate is to reliably assign information about the subject, i. e., the owner, of a public key. The owner may be identified by its name, or email address in case of a person, or by its network name (DNS name), or IP address of a server. Additional information encodes usage information about the public key respectively the digital certificate, as validity period, and allowed key usages as user authentication, or email encryption. For device certificates, it is possible to encode the device manufacturer, the device model, and the serial number within a device certificate.

The most commonly used certificate format is ISO X.509 [12]. Figure 1 shows the format, and some exemplary fields. The main purpose of a digital certificate is to bind a public key (`Subject Public Key Info`) of an entity to the name of the entity (`Subject`). Additional information as the validity period, the issuer, and usage restrictions can be included as well. X.509 certificates also support extensions, so that specific information can be included in addition.

When a digital certificate of a subject is validated by a communication peer, it is verified that the certificate has a valid digital signature of a trusted certification authority. It is

furthermore verified that the entries of the certificate match the intended usage, and that the certificate has not yet expired. It may also be verified whether the certificate has been revoked. A revocation check, see also subsection II-D below, may verify whether a given certificate is included in a certificate revocation list (CRL), or an online revocation status check may be performed using the open certificate status protocol (OCSP) [15]. In either case, at least partial online access to a PKI entity that is issuing certificates, and providing revocation information is needed at least from one component in an automation network, or cell. This component may further distribute the information within the automation cell.

C. Certificate Root Key

A digital certificate has to be validated before it is accepted. This includes a check whether the digital signature protecting the certificate is trusted. The standard approach is to use a set of trusted root certificates for certification authorities CA. A certificate is accepted if its signature chain can be verified back to a trusted root certificate. The root certificate may belong to a globally recognized CA, or to a local CA that is accepted only within an administrative domain, e. g., within a single operator network. If no PKI with CA is available, it is also possible to use self-signed certificates. This means that each certificate is signed with the private key associated with the public key contained in the certificate. Such certificates have to be configured as trusted in the same way as trusted root certificates, i. e., the (self-signed) certificates of trusted peers have to be configured explicitly. This requires to store the trusted peer information (root CA, or self signed certificates) in a secure manner, as this information is crucial for system security. A potential attack is the inclusion of an untrusted root certificate in the list of root certificates managed by a device. This attack is not specific to field devices. Some operating systems and web browsers also feature a certificate store containing a variety of root certificates that could be compromised. If an adversary would be able to introduce a new untrusted certificate into this root certificate store, he would compromise the system. Hence, the certificate store or certificate list has to be protected.

D. Certificate Issuance and Revocation

A digital certificate is issued by a certification authority (CA) of the public key infrastructure (PKI). This means that the certification authority creates the signed certificate for an entity, protected by the digital signature of the CA. It is common that the request to issue a certificate is received, and checked by a registration authority (RA), separating the checking, and the cryptographic functionality. A request to issue a digital certificate is sent typically using a certificate signing request (CSR) [16], using the simple certificate enrollment protocol (SCEP) [17], or by using enrollment over secure transport (EST) [18].

In cases where no PKI can be setup, also self-signed certificates are used. Here, an entity creates its own certificate, and signs it. The self-signed certificates, or a hash values (fingerprints) of the certificate, have to be configured on peers. This is practical only for small environments, due to the involved administration effort.

For industrial environments, an approach is to use pre-installed manufacturer device certificates on devices. These can be used to securely configure operational device credentials, and to protect certificate requests for operational device certificate to be used during operation.

A digital certificate can be revoked by the issueing CA. The most common approach is to use a certificate revocation list (CRL). A CRL is issued by a CA, indicating all certificates that have been issued, and later revoked by the CA. The drawback is that the current CRL has to be downloaded from the CA regularly, and that the size of a CRL can grow to large sizes being well suited for resource-limited IoT devices. An alternative is to use the OCSP protocol [15] to check the revocation status of a single certificate. This approach has the drawback that online connectivity is required.

In industrial environments, also short-lived certificates are used. The time of validity is set to a short time period so that revocation checks can be omitted [19].

E. Certificate Whitelisting

The basic concept of certificate whitelists is well-known. The underlying idea is to enumerate explicitly all authorized certificates. A certificate is validated successfully only if it is contained in the certificate whitelist. The whitelist may contain the certificates directly, or reference the certificates by their serial number, and issuer, by the certificate fingerprint, or by the public key. The latter avoids issuing a new whitelist, when a certificate is updated.

Such a certificate whitelist can be considered, and used also as an access control list that contains the certificates of all authorized subjects. Without using specific certificate extensions to encode different types of access, the different operations cannot be distinguished directly, however. Different certificate whitelists would have to be defined for different types of access. The configuration of the set of trusted root certificates is also a form of certificate whitelists. It is known to check whether the certificate of a communication peer is included in a certificate whitelist [20]. Also, the Microsoft Digital Rights Management License Protocol is using a certificate whitelists [21].

As these certificate whitelists have been used as a proprietary means for configuring a list of trusted certificates, or to be more precise a *set* of trusted certificates, the approach has been rather limited as general means for certificate management.

Other examples can be given through the pinning of certificates, which is also often done using CWL-like list. In contrast to the CWL approach described in this paper, the "classical" pinning takes the certificate from the very first connection as secure. It merely ensures that connections established afterwards are always verified against the list of "first connection certificates". The protocol secure shell (SSH) [7] is an example for relying on this approach: The server key

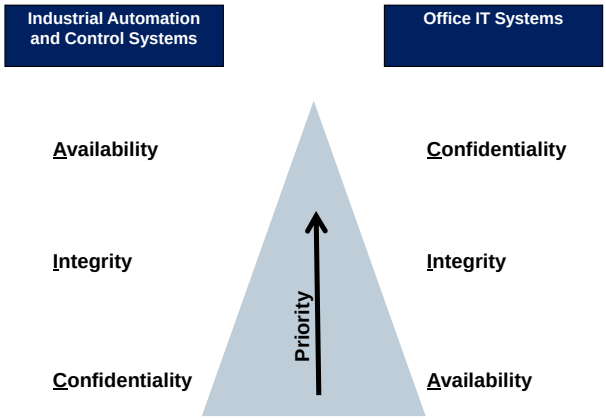


Fig. 2. Inverted CIA Pyramid

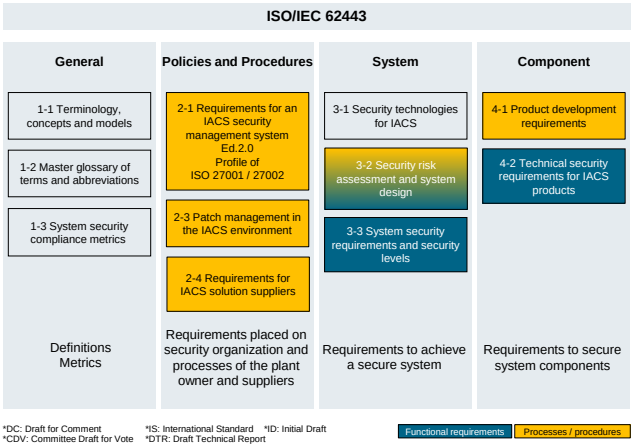


Fig. 3. Parts of ISO/IEC 62443

of the first connection is stored. There exist browser plug-ins that provide the same functionality for SSL/TLS-protected connections, like Certificate Patrol [22].

III. Security Standard IEC62443 for Industrial Automation Control Systems

Industrial automation control systems (IACS) monitor, and control automation systems in different automation domains. As networked automation control systems are exposed to external systems, they have to be protected against attacks to prevent manipulation of control operations. Krotofil and Gollmann have summarized research in the area of industrial control systems security [23]. Attacks have occurred in real world, see [24] reporting on a cyber attack where a steel mill could not be shut down correctly, causing severe damages.

The three basic security requirements are confidentiality, integrity, and availability. They are also named "CIA" requirements. Figure 2 shows that in common information technology (IT) systems, the priority is "CIA". However, in automation systems, also called operation technology (OT), or industrial IT, the priorities are just the other way round: Availability has typically the highest priority, followed by integrity. Confidentiality is often no strong requirement for control communication. Shown graphically, the CIA pyramid is inverted (turned upside down) in automation systems.

Specific requirements, and side conditions of industrial automation systems like high availability, planned configuration (engineering info), long life cycles, unattended operation, realtime operation, and communication, as well as safety requirements have to be considered when designing a security solution. The IT (information technology) security requirements defined in [10] can be mapped to different automation domains, including energy automation, railway automation, building automation, process automation, and others.

The security standard ISO/IEC 62443 [10] defines security for industrial automation control systems. Several parts have been finalized, or are currently in the process of being defined, see Fig. 3. The different parts cover common definitions, and metrics, requirements on setup of a security organization, and processes, defining technical requirements on a secure system, and to secure system components.

A complex automation system is structured into zones that are connected by so-called "conduits". For each zone, the targeted security level (SL) is derived from a threat and risk analysis. The threat and risk analysis evaluates the exposure of a zone to attacks as well as the criticality of assets of a zone. While IEC 62443-3-2 defines security levels, and zones for the secure system design, IEC 62443-3-3 describes the requirements to comply with a dedicated security level in an abstract way, not prescribing the actual implementation.

Four security levels have been defined, targeting different categories of attackes:

- SL1: Protection against casual, or coincidental violation
- SL2: Protection against intentional violation using simple means, low resources, generic skills, low motivation
- SL3: Protection against intentional violation using sophisticated means, moderate resources, IACS specific skills, moderate motivation
- SL4: Protection against intentional violation using sophisticated means, extended resources, IACS specific skills, high motivation

For each security level, IEC62443 part 3-3 defines a set of requirements. Seven foundational requirements group specific requirements of a certain category:

- FR 1 – Identification and authentication control
- FR 2 – Use control
- FR 3 – System integrity
- FR 4 – Data confidentiality
- FR 5 – Restricted data flow
- FR 6 – Timely response to events
- FR 7 – Resource availability

The security standard ISO/IEC62443 [10] part 3.3 states several requirements affecting device authentication under the

group FR1 "identification and authentication control". Some most relevant requirements are summarized here:

- SR1.2 — Software process and device identification and authentication: All devices, and software processes shall be possible to be identified, and authenticated. This requirement is relevant from security level SL2, and higher. While in SL2, group-, or role-based identification, and authentication is permitted, for SL3, and SL4, a unique identification, and authentication of devices is required.
- SR1.5 -– Authenticator management: Authenticators are credentials used to authenticate users, devices, or software processes. They have to be initialized, and refreshed. Initial authenticators shall be possible to be changed. The requirement is relevant for SL2, SL3, and SL4. For SL3, and SL4, a hardware mechanism is required to protect authenticators.
- SR1.8 -– Public key infrastructure (PKI) certificates: When a PKI is used, it shall be operated according to commonly accepted best practices, or public key certificates shall be obtained from an existing PKI. The requirement is relevant for SL2, SL3, and SL4.
- SR1.9 — Strength of public key authentication: When digital certificates are used, the certificate, the certificate path, and the certificate revocation status have to be checked. In SL3, and SL4, private keys have to be protected using a hardware-based mechanism.

These requirements have to be fulfilled while respecting side-conditions on high availability, and keeping safety-critical control networks closed. These imply that a control system should continue to operate locally, independently from any backend systems, or backend connectivity. Local emergency actions, as well as essential control functions shall not be hampered with by security mechanisms.

IV. Certificate Management and Validation Using Certificate Whitelists

The setup, and operation of a public key infrastructure has shown to require significant effort, and costs. This has been a limiting factor for the practical usage of public key cryptography. Ongoing standardization activities define the technological basis for simpler usage of public key cryptography for industrial automation environments, and the future Internet of Things.

While a certificate whitelist has been used so far as proprietary means for configuring some digital certificates as trusted, a certificate whitelists format is currently being standardized for the smart energy grid environment. It has been acknowledged that the application of certificate whitelists in restricted environments supports the long term administration of security parameters. Hence, standardizing the format is the next consequent step to ensure interoperability of different vendor products.

A certificate whitelist is a data structure containing respectively referencing a set of trusted, or authorized digital certificates. A certificate can be referenced by its serial number, and issuer, or by a fingerprint of the certificate (hash value). The certificate whitelist is signed using a whitelist root key of trust (WROT). A Certificate White List is also referenced as Certificate Authorization List, e. g., by the ITU-T X.509 group.

A certificate is validated successfully if it is contained in a corresponding certificate whitelist. Further checks on the contents of the certificate as the name of the subject, the certificate extensions, and the certificate signature are performed in the usual way.

Certificate whitelists can be used with certificates issued by a CA, or with self-signed certificates. A common technological basis is provided for smaller environments using self-signed certificates as well as environments using a PKI for issuing certificates. So, a smooth migration from self-signed certificates to a local PKI, and even towards global PKI is provided. Whitelists are advantageous also in the case when device certificates, having a very long validity period of, e. g., 30 years, are used. In this case, such a long-lived device certificate is accepted only if the certificate is valid, and if, in addition, it is included in a certificate whitelist.

A certificate can be revoked easily by *not* including it anymore in the certificate whitelist. This supports that the requirement SR1.9 of ISO/IEC62443 [10] part 3.3 is fulfilled, without having to rely on backend security servers that would be required for CRL-based or OCSP-based revocation checks. However, it is also possible to check the certification revocation status using certificate revocation lists [12], or using the online certificate status protocol OCSP [15] in addition.

1) Standardization Activities: Currently, ongoing standardization activities performed by the working group IEC TC 57 WG15, standardizing ISO/IEC 62351 [25] in alignment with ITU-T X.509 [12] define the usage of certificate whitelists for energy automation systems. Currently, a format is defined for a certificate whitelist. Figure 4 shows a recent proposal for a certificate whitelist. It is based on the format of a certificate revocation list CRL, but its assigned type (`CertificateWhiteList`) distinguishes it from a CRL. Also, the intended scope of a certificate whitelist is defined by a specific attribute `scope`. It allows a client to verify whether a certain certificate whitelist has in fact been intended for a specific purpose. For example, the IP addresses, or DNS names of devices for which the whitelist is intended to be used, can be included.

The target scope of a certificate whitelist can be explicitly encoded in a certificate whitelist. Therefore, a certificate whitelist cannot be used unintentionally for a different purpose as the intended purpose at time of compilation. Certificate whitelists can be compiled once during as part of engineering. Alternatively, end devices can pull a certificate whitelist from a whitelist certificate server in defined time intervals. The CWL can also be pushed to the field devices.

A digital certificate may be intended to be used only within a certificate whitelisting environment. To ensure that a certificate is in fact validated successfully only together with a corresponding whitelist, it is possible to include a corresponding extension in the certificate. The extension marks it explicitly

```
CertificateWhiteList  ::=  SEQUENCE  {
     tbsCertWhiteList     TBSCertWhiteList,
     signatureAlgorithm   AlgorithmIdentifier,
     signatureValue       BIT STRING
}

TBSCertWhiteList  ::=  SEQUENCE  {
   version      Version OPTIONAL,
                -- if present must be v1
   signature    AlgorithmIdentifier,
   issuer       Name,
   thisUpdate      Time,
   nextUpdate      Time OPTIONAL,

   scopedList      SEQUENCE OF SEQUENCE {
      scope        ScopeConstraints,
                   -- geographic,organizational
      authorizedCertificates    SEQUENCE OF SEQUENCE  {
         fingerprint     AlgorithmIdentifier, -- for FP creation
         certIdentifier::== CHOICE {
         serialCert      [0] CertificateSerialNumber,
         fingerprintCert [1] OCTET STRING -- FP of certificate
         fingerprintPK   [2] OCTET STRING -- FP of public key
   }

   certificateIssuer     Name OPTIONAL,
   cwlEntryRestriction   [0] EXPLICIT Extension OPTIONAL
                         -- further restrictions of cert. usage
   }
   }

   cwlExtensions [0] EXPLICIT Extensions OPTIONAL
                 {- for future use
}
```

Fig. 4. Certificate Whitelist Format [25]

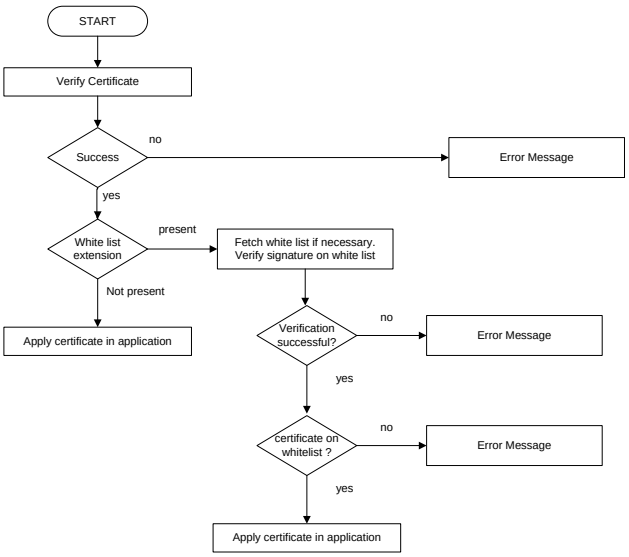


Fig. 5. Validation of a Certificate with Certificate Whitelisting

to be accepted only if it is included in a certificate whitelist. A corresponding certificate extension is currently defined in the scope of certificate management by ISO/IEC 62351 [25].

The validation of a certificate depends on whether it contains a certificate whitelist extension. Figure 5 shows the relevant checks. If a certificate includes the whitelisting extension, it is required that the corresponding whitelist is available, and that the certificate is in fact included in the whitelist.

V. Managed Certificate Whitelists

The introduction of certificate whitelisting implies the need for a management system for certificate whitelists. Managed certificate whitelists are a new approach for using public key cryptography in a practical, efficient, and effective way. It is particularly suited for systems with well-known set of devices, and their communication relationships, as it is common for networked automation systems. As the management of whitelists can be fully automated, it scales well to larger number of devices, although due to the increasing size of whitelists the targeted application environment is characterized by a number of devices within a range up to some 100 to some 1000 devices. It integrates well within existing industrial workflows for installing, or exchanging devices, as device configuration databases are kept up-to-date within automation systems. So, the information that is required to generate updated certificate whitelists is already available. Once certificate whitelists have been generated, and installed on the target devices, the target devices can operate autonomously even if the security infrastructure is not available. This is an important property for automation environments with high availability requirements to ensure that the automation system can continue to operate even if backend systems are temporarily unavailable.

A. Whitelist Generation and Distribution

The basic concept for automatic whitelist management is rather straightforward. Engineering information about the devices, and their communication relationships within a networked automation system is available in common automation systems. Several purpose-specific – and also device-specific if needed – certificate whitelists are generated automatically using this engineering information. The whitelists are distributed to the target devices using remote configuration protocols. For example, secure copy scp [7], HTTPS [26], or OPC-UA [27] can be used to distribute configuration files securely to the target devices.

Figure 6 shows the main components involved in the automatic management of certificate whitelists. A central device management component accesses a device database including all registered devices of a networked automation system, and their associated device certificates. Using automation system configuration data, the communication relationships are determined. This gives the list of the components installed in the automation system, and their communication links. Based on this information, certificate whitelists comprising the relevant certificates, can be compiled for the different communication purposes as automation control communication, supervisory control communication, remote service access, and diagnostic access. Depending on policy, device-specific certificate whitelists can be compiled, or certificate whitelists for defined purposes, and target device classes. The certificate whitelists are created, and provided to a device management system that configures the relevant certificate whitelists on the target devices. As important difference to a certification revocation list CRL, a certificate whitelist will usually be provided, and be signed by the operator, not by the certification authority

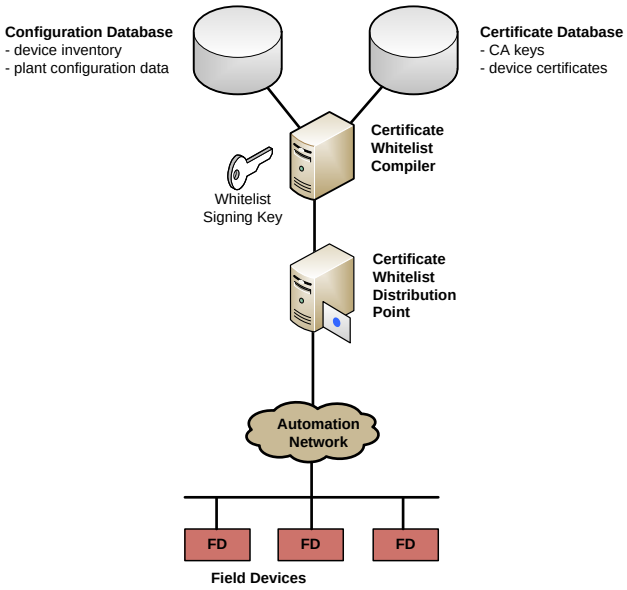


Fig. 6. Certificate Whitelist Management System

(CA). As the whitelist is a digitally signed data structure, the corresponding private key has to be protected, for example by using a smart card, or a hardware security module (HSM). However, note that while the digital certificates are signed usually by a certification authority (CA), the whitelist will by managed, and digitally signed, by the operator of the automation system. This has the advantage that an automation system operator can use managed certificate whitelists easily with certificates issued by different CAs, and even with long-lived, or self-signed certificates.

For networked automation systems with a typical size of some 100 to some 1000 devices, such a certificate management system based on whitelisting provides several advantages for the application in real-world industrial usage scenarios: A local PKI, long-lived, or even self-signed certificates can be used, so that a deployment with a very limited security infrastructure is possible. For the operation of the automation system, no continous reachability, or availability of the whitelisting security infrastructure is required. So, the availability of the automation system availability does not depend on the availability of the security infrastructure. A commonly availably device management infrastructure can be extended easily for automatically creating, and distributing certificate whitelists. It is possible to use a certificate whitelist only for authentication. Authorization checks would then be performed in addition, e. g., by checking an access control list. However, a certificate whitelist can be used directly as access control list as well. Different certificate whitelists would be configured for the different types of access (e. g., control communication, service access, diagnosis). The current proposal for a CWL structure considers this by supporting the encoding of a list of lists. Moreover, within the CWL, further certificate usage restrictions may be encoded. One example is the definition of dedicated applications, or communication protocols, which are allowed to utilize a dedicated certificate. Using this approach, the communication peer could refuse to accept a certificate included on the CWL if it is not associated within the CWL with the currently used communication protocol.

This approach has the advantage that no separate identity, and access management infrastructure is needed, and that access control decisions can be performed by a field device when the backend systems are not available. These properties make certificate whitelisting a very interesting approach for managing digital certificates in typical industrial automation systems.

B. Example Usage Scenarios

Typical workflows in industrial automation systems are the initial installation, the replacement, and removal of devices. As device configuration databases are already maintained as part of these workflows, the information for updating certificate whitelists is available without any extra effort required from the service personnel.

The certificate whitelist management system is triggered by a change in the configuration database. When a change in the configuration has been detected by the certificate whitelisting system, the generation of updated certificate whitelists is started. This happens preferably when a service mode of the automation system is terminated. The generated certificate whitelists are deployed automatically on the affected target devices using remote service access protocol, e. g., HTTPS [26], scp [7], or OPC-UA [27].

VI. Application Within Energy Automation Systems

The general approach of using managed certificate whitelists as described in the previous section can be applied for energy automation systems (smart grid). Figure 7 shows a substation automation system. A substation typically transforms voltage levels, and includes power monitoring, and protection functions. Figure 7 shows separate network zones of the substation communication network. The field devices that perform the actual field level functionality of monitoring, and acting on the electric power are called intelligent energy devices (IED). They are monitored, and controlled by a substation controller, realizing a realtime automation system. Energy automation protocols are defined by the standard IEC 61850 [28] which specified the Generic Object Oriented Substation Events (GOOSE) protocol, realizing the realtime communication with transfer requirements smaller than microseconds by utilizing multicast Ethernet connections between the field devices. Additional network zones are available for local, and remote service access, for integrating intelligent field devices with serial interfaces, and for support functions (file server, historian server for logging, remote access server, terminal server). A substation is connected to the utility communication network providing backend services like supervisory control, and data

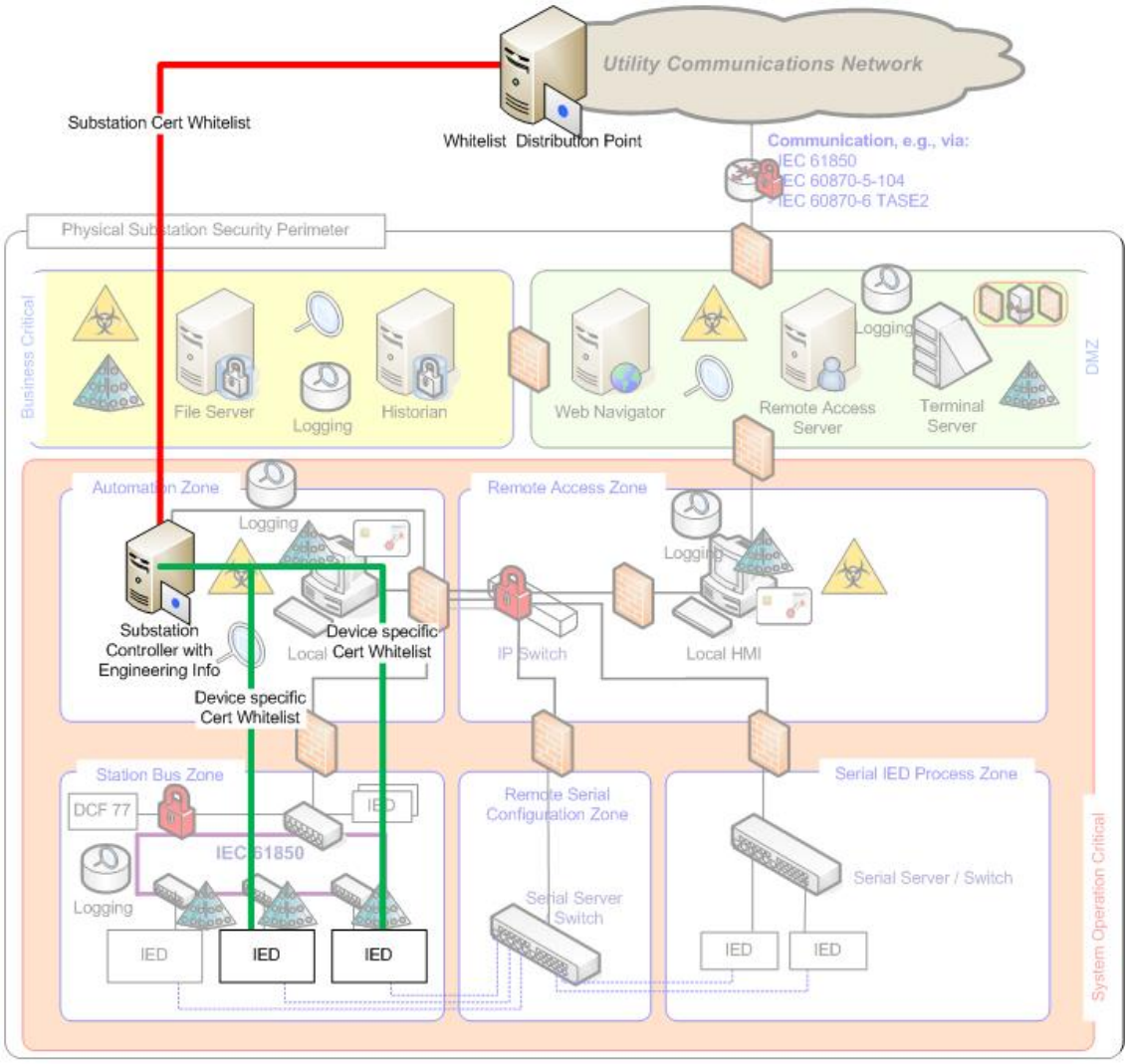


Fig. 7. Managed Certificate Whitelisting in Energy Automation Substations

acquisition (SCADA). Firewalls are used to control the traffic flow between zones.

A hierarchical creation, and distribution of certificate whitelists to a substation may be realized in the following way: A utility operator creates a substation-specific certificate whitelist (substation cert whitelist) based on the engineering information for this substation, and distributes it to the substation controller. This certificate whitelist contains the subset of certificates that are relevant for the substation. The specific substation is encoded in the CWL by the scope restriction. Using engineering information that is available at the substation controller, the substation controller creates device-specific certificate whitelists for the field devices, i. e., intelligent energy devices (IED), of the substation. The device-specific certificate whitelists are configured by the substation controller on the different IEDs.

An alternative approach would be to compile a CWL for a substation, and to distribute this CWL to all components in the substation via the substation controller. Through the engineering information, each IED will only communicate with other IEDs by means of the engineering data, and the CWL. This means that the access control decision is made by an IED by checking both the CWL, and the engineering information. This saves the additional effort for creating device specific CWLs, but has the disadvantage that each IED needs to search a larger CWL, and has to check two pieces of configuration information separately. It is a validation perfomance decision which approach is more appropriate in a target environment. The generic definition of CWLs allows for both approaches.

A further usage scenario for certificate whitelisting within energy automation systems would be the integration of decentralized energy resources. Here, a smart grid operator may realize a (managed) certificate pinning by using certificate whitelists. A smart grid operator would define which certificates are acceptable by including these certificates in a whitelist. Thereby, the smart grid operator would use certifi-

cate whitelists to restrict the set of certificates issued by a larger PKI. The possibility to misuse broken certificates, or CAs is reduced as the set of accepted certificates is limited.

VII. Conclusion and Outlook

Industrial automation control systems (IACS) monitor, and control automation systems in different automation domains, e. g., energy automation, railway automation, or process automation. As networked automation control systems are exposed not only to local attacks, but also to attacks originating from external systems, they have to be protected against attacks to prevent manipulation of control operations. Security requirements for automation systems have been defined by the industrial security standard ISO/IEC62443 [10], distinguishing four security levels.

The automation communication can often be protected using standard security protocols. Asymmetric cryptographic keys, and corresponding device certificates are used as symmetric keys would not scale well for the huge number of involved devices. Main considerations are the demand for extremely high system availability, requiring that the automation system can continue to operate in an autonomous island mode, and the fact that many automation systems are setup as separate network segments that have no, or only limited connectivity with general office networks, or even the public Internet.

This paper described a new approach for the practical management of device certificates for field-level automation devices, based on certificate whitelists. The fact that automation systems are typically engineered, e.g., that the communication relations are known up front, can be leveraged for automated certificate and access management. The basic concept of certificate whitelists is well-known. The underlying idea is to enumerate explicitly all authorized certificates. A certificate is validated successfully only if it is contained in the certificate whitelist. The whitelist may contain the certificates directly, or reference the certificates by their serial number, and issuer, by the certificate fingerprint, or by the public key. Such a certificate whitelist can be considered, and used also as an access control list that contains the certificates of all authorized subjects. Without using specific certificate extensions to encode different types of access, the different operations cannot be distinguished directly, however. Different certificate whitelists would have to be defined for different types of access.

Explicitly designating trusted certificates in certificate whitelists has been recently put forward within standardization for industrial energy automation communication [25]. It promises to provide a cost-efficient, easily deployable, and operable approach for digital device certificates even if self-signed certificates are used. It is intended for mid-sized industrial automation domains, while providing a migration path to more flexible PKI, and access management structures. It allows in particular to avoid the usage of simple manually configured pre-shared secrets, which would be difficult to migrate to more complex, and managed security infrastructures that are expected to be advantageous for large scale deployments. Its application is beneficial also in other industrial automation domains, e. g., railway automation, where very high availability requirements have to be fulfilled. Certificate whitelisting enables that a local control system can continue to operate autonomously when backend systems are not accessible for a certain time. They provide a way to fulfill the requirement for certificate revocation check, posed by industrial security standard ISO/IEC 62443 part 3.3 [10] independently from backend security servers (e. g., servers for identity, and access management, distribution points for certificate revocation lists, or online certificate status servers).

The usage of certificate whitelisting can be supported with automatic whitelist generation, and distribution. A format for certificate whitelists is currently proposed for standardization in ITU-T X.509, and for application in ISO/IEC 62351 in the context of key management in power automation. Specific extensions can mark a certificate explicitly for being used only in combination with a certificate whitelist.

Several additional extensions may be introduced in the future. It is possible to indicate usage restrictions within a certificate whitelist associated with a certain certificate entry. This could be used to limit the authorized usage of a certificate on a certificate-by-certificate basis. Certificate whitelists may be encoded efficiently by including matching criteria of included certificates. Alternatively to the explicit enumeration of certificates, a filter can be included in a certificate whitelist that defines matching criteria of included certificates, i. e., that defines required properties of certificate fields. A Bloom filter [29] may be used, combined with a check on false match. Bloom filters are a probabilistic data structure for membership queries which allow for an efficient encoding, but for which a wrong positive match may occur. As the set of all issued certificates is known in typical usage scenarios, a checking for a false match is easily possible. Also, certificates can be designated within a whitelist. Also, a PKI gateway can be deployed for secure interworking with external network domains using a standard public key infrastructure.

Also, the logical combination of multiple certificate whitelists is possible in general. The general concept of structured definition of access control policies by logically combining partial access control policies has been described, e. g., by [30]. A combination of certificate whitelists may be advantageous for instance in an inter-substation communication scenario. Here, a first certificate whitelist may be provided for the substation internal communication, and a second one for inter-substation communication. The final certificate whitelist for each purpose may be defined by a logical combination of whitelists to ease the certificate whitelist administration, and the handling for the field device. This might be done by logical OR, AND, or XOR combinations of the certificate whitelists. This logical combination can be realized in different ways: The field devices themselves can check against multiple certificate whitelists. A logical expression is configured that defines the logical combination of the certificate whitelists to be applied. As the defined certificate whitelist structure shown in Fig. 4 allows the encapsulation of multiple certificate

whitelists within a single data structure, an enhancement of this data structure could indicate the logical combination of the whitelist entries using the extension option.

A further alternative would be the preparation of device specific certificate whitelists by a centralized infrastructure component that determines the result of the logical combination of different certificate whitelists before distributing the actual certificate whitelist to the end points. This puts more effort on the centralized component, but keeps the effort low for the field device. The assumption here is that the certificate whitelist for a single endpoint is rather short compared to substation wide certificate whitelists containing all allowed (engineered) combinations of communication associations.

The structure defined in Fig. 4 also allows using different matching criteria for the certificate. While the serial number, and issuer, or the fingerprint are straight forward, the utilization of the public key fingerprint provides another degree of freedom. This approach allows even for updating certificates (assumed the public key stays the same) without changing the CWL. This decouples the certificate life cycle management from the access security policy management of certificates in automation environments.

References

[1] R. Falk and S. Fries, "Managed Certificate Whitelisting – A Basis for Internet of Things Security in Industrial Automation Applications," in *Proceedings of the 8th International Conference on Emerging Security Information, Systems and Technologies (SECURWARE), November 16–20, 2014, Lisbon, Portugal*. ThinkMind, Nov. 2014, pp. 167–172.

[2] IEEE 802.11, "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." [Online]. Available: http://standards.ieee.org/about/get/802/802.11.html [accessed: 2015-01-20]

[3] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Barners-Lee, "Hypertext Transfer Protocol – HTTP/1.1," 1999, Internet Request for Comments RFC2696. [Online]. Available: https://tools.ietf.org/html/rfc2696 [accessed: 2015-01-20]

[4] IEEE 802.1X-2010, "IEEE Standard for Local and metropolitan area networks–Port-Based Network Access Control," . [Online]. Available: http://standards.ieee.org/findstds/standard/802.1X-2010.html [accessed: 2015-01-20]

[5] C. Kaufmann, P. Hoffman, Y. Nir, and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)," Sep. 2010, Internet Request for Comments RFC5996. [Online]. Available: https://tools.ietf.org/html/rfc5996 [accessed: 2015-01-20

[6] S. Kent, and K. Seo, "Security Architecture for the Internet Protocol," Dec. 2005, Internet Request for Comments RFC4301. [Online]. Available: https://tools.ietf.org/html/rfc4301 [accessed: 2015-01-20]

[7] T. Ylonen, and C. Lonvick, "The Secure Shell (SSH) Protocol Architecture," Jan. 2006, Internet Request for Comments RFC4251. [Online]. Available: https://tools.ietf.org/html/rfc4251 [accessed: 2015-01-20]

[8] Netscape, "SSL 3.0 specification," Nov. 1996. [Online]. Available: http://web.archive.org/web/20080208141212/http://wp.netscape.com/eng/ssl3/ [accessed: 2015-01-20]

[9] T. Dierks, and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2," Aug. 2008, Internet Request for Comments RFC5246. [Online]. Available: https://tools.ietf.org/html/rfc5246 [accessed: 2015-01-20]

[10] ISO/IEC 62443, "Industrial Communication Networks -– Network and System Security," 2014, IEC TC57. [Online]. Available: [accessed: 2015-01-20]

[11] IEC IECEE, "IEC System of Conformity Assessment Schemes for Electrotechnical Equipment and Components (IECEE)," 2015, . [Online]. Available: http://www.iecee.org/ [accessed: 2015-01-20]

[12] ITU-T X.509, "X.509 Information technology – Open Systems Interconnection -– The Directory: Public-key and attribute certificate frameworks," 2012, version 3 corrigendum 3. [Online]. Available: http://www.itu.int/rec/T-REC-X.509-201210-S!Cor3/en [accessed: 2015-01-20]

[13] D. Cooper, S. Santesson, S. Farrel, S. Boeyen, R. Housley, and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," May 2008, Internet Request for Comments RFC5280. [Online]. Available: https://tools.ietf.org/html/rfc5280 [accessed: 2015-01-20

[14] J. Buchmann, E. Karatsiolis, and A. Wiesmaier, "Introduction to Public Key Infrastructures," 2013.

[15] S. Santesson, M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP," Jan. 2013, Internet Request for Comments RFC6960. [Online]. Available: https://tools.ietf.org/html/rfc6960 [accessed: 2015-01-20]

[16] M. Nystrom, and B. Kaliski, "PKCS #10: Certification Request Syntax Specification Version 1.7," Nov. 2000, Internet Request for Comments RFC2986. [Online]. Available: https://tools.ietf.org/html/rfc2986 [accessed: 2015-01-20

[17] M. Pritikin, A. Nourse, and J. Vilhuber, "Simple Certificate Enrollment Protocol ," Sep. 2011, Internet Draft draft-nourse-scep-2 (work in progress). [Online]. Available: http://tools.ietf.org/html/draft-nourse-scep-23 [accessed: 2015-01-20

[18] M. Pritikin, P. Yee, and D. Harkins, "Enrollment over Secure Transport," Oct. 2013, Internet Request for Comments RFC7030. [Online]. Available: https://tools.ietf.org/html/rfc7030 [accessed: 2015-01-20

[19] S. Fries and R. Falk, "Securely connecting Electric Vehicles to the Smart Grid," *International Journal On Advances in Internet Technology*, vol. 6, no. 1 and 2, pp. 57–67, 2013, ISSN: 1942-2652.

[20] eTutorials.org, "C/C++ Secure Programming – Chapter 10.9 Using a Whitelist to Verify Certificates," 2014, eTutorials.org. [Online]. Available: http://etutorials.org/Programming/secure+programming/ [accessed: 2015-01-20]

[21] Microsoft, "Digital Rights Management License Protocol – Retrieving Revocation Data from the Enrollment Server," 2014. [Online]. Available: http://msdn.microsoft.com/en-us/library/dd644914.aspx [accessed: 2015-01-20]

[22] CertPatrol, "Certificate Patrol 2.0," 2015. [Online]. Available: http://patrol.psyced.org/ [accessed: 2015-01-20]

[23] M. Krotofil and D. Gollmann, "Industrial Control Systems Security: What is Happening?" in *Proceedings of the 11th IEEE International Conference on Industrial Informatics (INDIN), 29-31 July 2013, Bochum*. IEEE, Jul. 2013.

[24] heise news, "BSI-Sicherheitsbericht: Erfolgreiche Cyber-Attacke auf deutsches Stahlwerk," 2014. [Online]. Available: http://www.heise.de/newsticker/meldung/BSI-Sicherheitsbericht-Erfolgreiche-Cyber-Attacke-auf-deutsches-Stahlwerk-2498990.html?view=print [accessed: 2015-01-20]

[25] ISO/IEC 62351, "Power Systems Management and Associated Information Exchange Data and Communication Security," 2014, IEC TC57. [Online]. Available: http://tc57.iec.ch/index-tc57.html [accessed: 2015-01-20]

[26] E. Rescorla, "HTTP Over TLS," 2000, Internet Request for Comments RFC2818. [Online]. Available: https://tools.ietf.org/html/rfc2818 [accessed: 2015-01-20]

[27] OPC Foundation, "OPC Unified Architecture Specification Part 1: Overview and Concepts, Release 1.02," Jul. 2012. [Online]. Available: http://www.opcfoundation.org/ua/ [accessed: 2015-01-20]

[28] ISO/IEC 61850, "IED Communications and Associated Data Models in Power Systems," 2014, IEC TC57. [Online]. Available: http://tc57.iec.ch/index-tc57.html [accessed: 2015-01-20]

[29] Wikipedia, "Bloom Filter." [Online]. Available: http://en.wikipedia.org/wiki/Bloom_filter [accessed: 2015-01-20]

[30] R. Falk, "A Method for Administering Access Rights in IT Systems [German: Eine Methode für die Verwaltung von Zugriffsrechten in IT-Systemen]," 2000, Dissertation, TU München.

Impacts on Database Performance in a Privacy-Preserving Biometric Authentication Scenario

Veit Köppen, Christian Krätzer
Jana Dittmann, Gunter Saake

Faculty of Computer Science
Otto-von-Guericke University
Email: [vkoeppen|kraetzer|
jana.dittmann|gunter.saake]@ovgu.de

Claus Vielhauer

Department of Informatics and Media
Brandenburg University of Applied Sciences
Email: vielhauer@fh-brandenburg.de

***Abstract*—Nowadays, biometric data are more and more used within authentication processes. Such data are usually stored in databases and underlie inherent privacy concerns. Therefore, special attention should be paid to their handling. We propose an extension to an existing privacy preserving similarity verification system. The Paillier scheme, being an asymmetric as well as additive homomorphic cryptography approach, enables signal processing in the encrypted domain operations. Amongst other modifications, we introduce a padding approach to increase entropy for better filling the co-domain. As a result, we combine the benefits of signal processing in the encrypted domain with the advantages of salting. The concept of verification of encrypted biometric data comes at the cost of increased computational effort in contrast to already available biometric systems. Nevertheless, this additional cost is in many scenarios justified by addressing that most currently available biometric authentication systems lack sufficient privacy protection. In our evaluation, we focus on performance issues of the privacy-preserving biometric authentication scheme with respect to database response time. The results presented for different evaluations on the influence of numbers of users, template sizes, and cryptographic key lengths show that the increase in effort required caused by our extensions is negligible. Furthermore, our improved scheme lowers the error rates attached as well as it reduces the amount of data that is disclosed in an authentication attempt. Our work highlights that user- and privacy-centric approaches to authentication have become feasible in the last few years. Modern schemes, as the one discussed in this paper, are not only efficient but also make the usage of data mining techniques in the domain of user tracking much more difficult.**

***Index Terms*—Database Security; Homomorphic Encryption; Privacy; Multi-Computer Scenarios; Database Performance; Biometric Authentication**

I. Motivation

Biometric data are more and more used in daily life. However, these data underlie privacy concerns by design, because these data are directly related to individuals. As a result, this may potentially be misused, e.g., by means of replay attacks, once accessible by malicious parties. Therefore, biometric data require protection mechanisms to take advantage of positive aspects of an authentication scheme. So, privacy-preserving biometric authentication is a requirement that comes into focus of databases, which form the core of any biometric system. In [1], the original conference article that is extended in this paper, we present a new approach for user authentication based on the assumption that encrypted data have to be stored and at the same time there is no logging information available.

Although data might be deleted from a database, it is possible to restore the information partly or even completely. Grebhahn et al. [2] present an approach for deleting data in a database whereas at the same time information could be completely recovered. Although new approaches exist to cover this information or even to improve the system for secure deletion [3], an overall security of traditional database management systems with respect to such information leakage cannot be guaranteed.

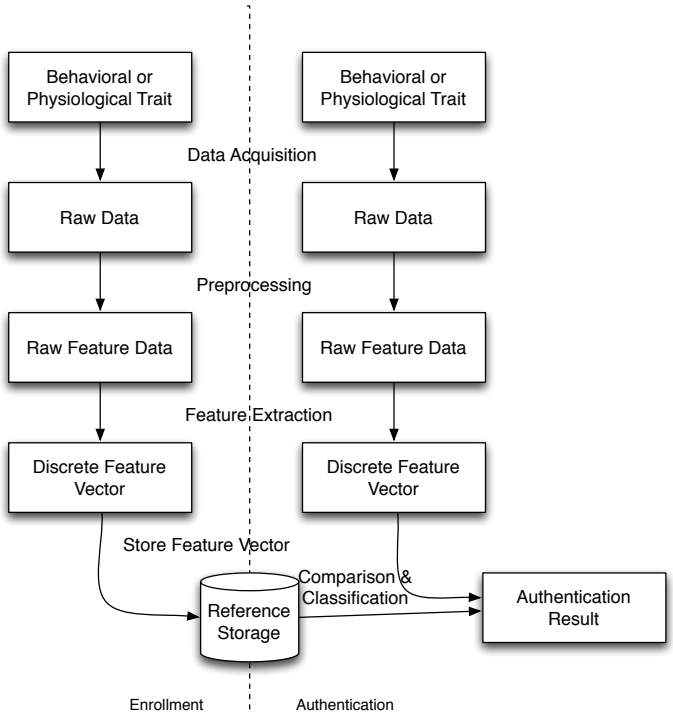


Figure 1. Enrollment and Authentication Pipeline

In a biometric authentication system, two phases are differentiated [4]. Firstly, a user has to create a specific biometric template. In practice, these templates are typically stored in a database. In order to store only required information, the data acquisition (e.g., by using sensors) is followed by a data preprocessing to filter out noise and non-related information of the raw data. Note that required information is often depicted in a feature space. Secondly, a feature extraction is applied, which is followed by a discretization of the feature values. Finally, the feature vector is stored. This phase is called **enrollment**. We show the basic steps in Figure 1 on the left side.

The second phase is called **authentication**, where a classification is required to declare an identity of the biometric features. We depict this pipeline on the right side of Figure 1. The first steps from data acquisition to the discrete feature vector should be applied in the same manners as in the enrollment phase. Otherwise, it cannot be guaranteed that the same properties are compared. However, the data for authentication are not stored. In the comparison step, if a one-to-one matching is performed, we call the authentication **verification** [4]. Another classification schema is **identification**, where a biometric discrete feature vector is compared to a set of templates from the database. In both schemes, usually a threshold is used to decide on the success of the authentication.

In case the threshold does not influence the comparison of templates, the result set of an identification can be the closest match, all, k-nearest, or ϵ-distance-neighbors. With these result-sets, further analyzes are possible, e.g., data mining or forensic investigations. Due to complexity, there are several optimization approaches possible. For instance, it is possible to use index structures within the database system for an enhanced data access. However, such index structures need to be carefully optimized for a multi-dimensional feature space, see for further details [5]. Another approach is to preserve privacy in the context of deletion in database index structures as described in [3].

Data mining enables users to detect patterns that are hidden in complex data. With the use of computational techniques, it is also possible to observe and identify relations in the context of privacy preserving scenarios, see for instance [6], [7], or [8].

The work presented in this paper is based on the paper [1] and extends the work as well as summarizes the main results. We present a methodology based on the Paillier cryptosystem [9] to improve user preferences with respect to authentication systems.

We present a cross-evaluation of the impact of homomorphic encryption for biometric authentication using a database within our evaluation section. The Paillier system is an asymmetric cryptographic scheme with additive homomorphic properties. With our new approach, both unique identifiers in our scheme (UID and FID, see Figure 5) need to be decrypted for every message. A disclosure of either the key is more unlikely, user-tracing becomes less likely, and the pad do not immediately reveal user content data.

The remainder of this paper is structured as follows: In Section II, we briefly describe the current state of the art regarding our new approach. In Section III, we present the architectural requirements for the application scenario of multi-computer involvement. Our extension of the secure similarity verification is given in Section IV. The evaluation of our approach regarding performance is part in Section V, where we show that response times are accompanied with a small computational effort for privacy preserving aspects. These findings are in line with theoretical considerations and assumptions. Finally, we conclude our results and give a short outlook in Section VI.

II. Background and Related Work

In this section, we present related work for preserving privacy in a biometric authentication context. As important factors, we concentrate on homomorphic encryption as well as deletion in database systems. The reason to focus here on homomorphic encryption instead of any other alternative cryptographic concept (see, e.g., [10]) is that this concept allows neglecting the crucial question of key provisioning.

With the majority of the established cryptographic schemes, the client either has to disclose a key to the database system (DBS) or, if such a disclosure is not allowed, has to perform the cryptographic functions itself. Both alternatives result in the transfer or registration of sensitive data items (either the keys or the data itself). With homomorphic encryption this is not necessary, because certain operations on the encrypted data can be performed by the DBS without possession of keys (see [10] and [11] for details).

Data security requirements target at properties of a system to protect data in a sufficient way. The main properties regarding data security are [12]:

- **Confidentiality** addresses the secrecy or prevention of unauthorized resources disclosure. In most practical cases, it refers to information, which needs to be treated secret from unauthorized entities.
- **Authenticity** is divided into two distinct aspects: Data origin authenticity and entity authenticity. Data origin authenticity is the proof of the data origin, genuineness, originality, truth, and realness. Entity authenticity is the proof that an entity has been correctly identified as originator, sender or receiver; it can be ensured that an entity is the one it claims to be.
- **Integrity** is the quality or condition of data objects being whole and unaltered, and it refers to their consistency, accuracy, and correctness.
- Given a set of entities and a resource, the resource has the property of **availability** if all entities of the set can successfully use the resource.
- **Non-repudiation** proves involved and third parties whether or not a particular event or a particular action occurred. The event or action can be, e.g., generation or sending of a message, receipt of a message, and submission or transport of a message.

The general security requirements for a biometric authentication system are summarized in [13]. Here, it is shown that all security aspects summarized in [12] become relevant for all enrollment and verification/identification related components as well as all data transitions between these. Privacy issues are mainly related to confidentiality, but require integrity, authenticity, availability, and non-repudiation of privacy related data. For each security aspect, a security level can also be introduced, e.g., ranging from non, low, up to high.

Within the domain of biometric authentication, data signals are often erroneous. The data are error prone due to noise within the acquisition process. This is the reason, why fault tolerance has to be carefully respected, too. Currently, only the One-Time-Pad approach can be considered as information-theoretically secure as long as the key is distributed securely.

Security plays a vital role due to different scenarios, in which an attack of personal data is imaginable. A differentiation of attacks can be made on a first level regarding passive or active attacks. The data stream between sender and recipient is not influenced in passive attacks. Therefore, only the reading of data is target for such attacks. Besides just reading data, a specialization is frequency analysis, where for instance for a substitution cipher an analysis of letter frequency is used to identify a mapping. Different extensions are applicable, e.g., frequency attacks or domain attacks [14].

Data mining and big data enable a high variety of data analytic techniques. In our biometric scenario, we hide user information to avoid a user tracking. However, it is possible to identify users with the help of log-files [15] or use pattern identification to even track anonymized users [16]. Furthermore, it is not necessary to to use as much information as possible, because a reduction of the multi-dimensional data spaces also reveals good patterns and deliver interpretable models [17].

In the concept of database performance, it has been shown that procedural extensions of modern database systems, such as Oracle PL/SQL and PostgreSQL PL/pgSQL, are very well suited for integrating cryptographic and steganographic functionality, e.g., [18]. This comes with a minimal performance overhead [19]. However, the authors have also shown that this approach comes with a large implementation and testing overhead. To this end, we consider this integration into databases as main focus.

In the following subsections, we first present background on the issue of template protection and secure deletion in databases, we also look into homomorphic encryption, which is followed by efficient biometric comparison in the encrypted domain.

A. Biometric Template Protection

A first idea to preserve privacy in the domain of biometric authentication is to store no direct data corresponding to personal information. In such a scenario, the templates are exchanged with a one-way hash function that is applied on the feature vectors.

As a result of the noise characteristics of biometric signals mentioned above, no cryptographic hash function can be applied directly for this task. Instead biometric hashes (or BioHashes, see [4]) are used. Those algorithms generate hash objects and supporting data required for the quantization operations used to stabilize the biometric input.

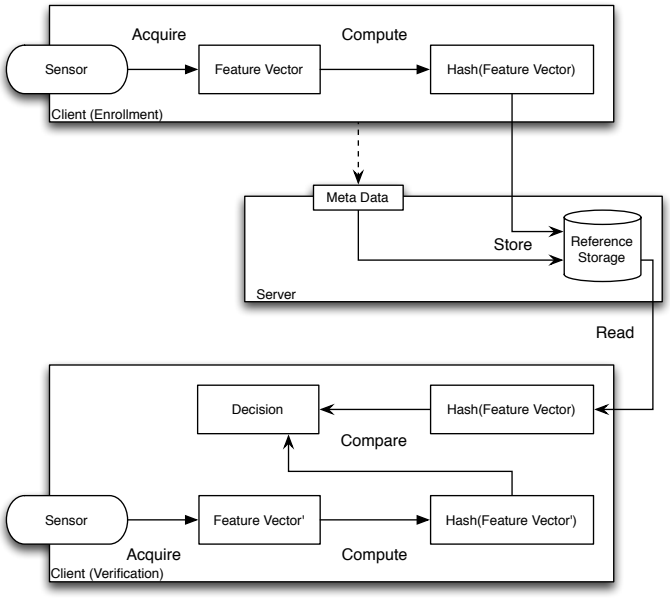


Figure 2. A Biometric Template Protection Example

In Figure 2, we present this approach for the verification of a user. In the database, only the hash values and user specific information such as the required interval matrix are stored. Meta data is also stored, to guarantee that future constructions perform in a comparable manner. The verification takes place at the client side, where the new biometric feature vector is handled as the comparable one with the same hash function. From the database the stored hash value and the corresponding user specific data are obtained and used for comparison.

Protection mechanisms for such Biometric reference systems exist since more than a decade; prominent examples are BioHashes [4], Fuzzy Commitment Scheme [20], and Fuzzy Vault [21]. For an overview on challenges for biometric template protection and further current protection schemes see [22]. All these established protection schemes require data to be compared in an unencrypted form, which leads to the threat of information leakage as discussed in Section I. Therefore, these mechanisms are not relevant for the work presented in this paper.

B. Secure Deletion in Databases

Databases can often reveal more information than intended. If an entry is deleted from the data collection, it is a mandatory step to avoid the data reconstruction afterward. Stahlberg et al. [23] and Grebhahn et al. [2] explain how data can be reconstructed from metadata or system copies. Furthermore, DBS specific data, such as index structures, can also be used for reconstruction of deleted data. This means, even if no data are left, the system inherent data structure can be used

to gain information from fully deleted data tuples. Therefore, privacy awareness for database tunings, as described in [3], is required for biometric DBS to guarantee data privacy, which is especially challenging for multi-dimensional data [24].

Apart from a possible reconstruction of previously erased data, saved data can reveal additional information. For instance, the amount of queries for a data tuple can give an idea about who that tuple belongs to. This kind of vulnerabilities of the confidentiality needs to be addressed early at the stage of the database layout. Not all security risks can be solved at this stage of the design, but a good database layout can indeed be the foundation of a secure system.

Our proposal here is to solve a prominent part of these confidentiality issues by not storing plain text items to the DBS and using homomorphic encryption to solve the key provisioning issue (i.e., the need for the system to have access to crypto keys).

C. Homomorphic Encryption

Homomorphic encryption is used to perform for asymmetric encryption schemes data operations on the cipher text, which have a corresponding operation on plain text data. In homomorphic encryption, operations op^* can be performed on encrypted data that are equivalent to operations op on the plain text. This means that the following formula holds:

$$op(x) = decryption\left(op^*(encryption\left(x\right)\right). \quad (1)$$

In such a case, the mapping is structure preserving. The operations op and op^* depend on the cryptosystem. There exist additive and multiplicative homomorphic cryptosystems. Gentry [25] proves the existence of a fully homomorphic encryption scheme having additive as well as multiplicative properties. So, it is possible to perform certain operations on data without possessing a decryption key. However, such systems require high computational effort as well as a translation of required operations on the functions provided by the homomorphic encryption scheme at hand (here the Paillier scheme [9]. In this paper, we make use of homomorphic encryption to perform operations for authentication in an encrypted domain. The basic idea for our work is derived from the work of Rane et al. as summarized in the next subsection.

D. Verification of Homomorphic Encrypted Signals

Rane et al. [11] [26] developed an authentication scheme with adjustable fault tolerance. This is especially important for noisy sensor data. Due to error correction and similarity verification, Rane's method can be applied for a wide range of biometric traits.

In their application, three participants are involved for a multi-computer scenario. Whereas the first user provides the biometric signals, the second involved user acts as the central storage server for all biometric templates. The third user is responsible for verification. However, this user is seen as vulnerable and therefore, she is not allowed to query the database system (DBS). Despite the fact that we also use this three participant setup for our evaluations, we present alternative scenarios in Section III.

III. Architecture for Privacy-Preserving Authentication

In a general authentication setup, there are two instances that have to share information with each other. There is a participant using a sensor to authenticate a claimed identity on the one side. On the other side, there is a reference DBS containing all enrolled data of all registered users. The DBS is considered to be semi-trustworthy, which means the data in this system shall never be available to the database holder without any kind of restriction or encryption. For that reason, a system allowing database authentication without revealing any information to the database holder needs to be applied. Furthermore, it has to be impossible to decrypt data without having the secret key. The solution used in this paper to address this issue is the use of homomorphic encryption.

Here, we use the Paillier crypto system as described in [9]. We slightly extend this scheme with the inclusion of user-definable key lengths for the purpose of the performance evaluations presented in Section V.

In Figure 3, we present a simplified pipeline of a verification process. For this paper, we consider this process layout as a standard pipeline. Note, in this scenario, a compromised DBS administrator could keep track of the order of enrolled employees and therefore, a sequential ID has to be avoided. This is also conceivable for timestamps and other metadata. So, it is inevitable to disable any logging of enrollment steps.

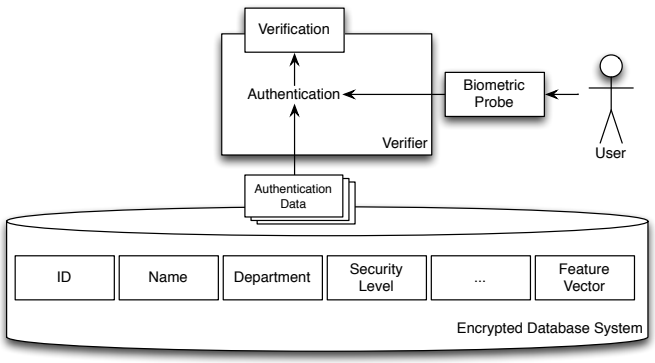


Figure 3. Authentication Process with Encrypted Database, adapted from [27]

In general, there are three major approaches with a different number of participants to be considered for a setup. First, there is a setup consisting of **two participants**. A participant holds sensor and private key and a non-trusted domain holding the data that are encrypted with the public key. The data can never be decrypted on the server side and therefore, need to be sent to sensor side for authentication. The two participants approach requires a secure layer (prospective two+ participants approach) to become trustworthy. This layer would be able to perform black box operations without revealing any information to the database holder or the user.

The second approach is the **three participants** approach from [11], [26], which is summarized in Figure 3. This approach consults a third member, called the verifier, which is

deemed semi-trustworthy as well. The new member shall gain as little information as possible. For that reason, the Paillier cryptosystem [9] is used for the instantiation of this approach within this paper.

The last major setup considered here is a multi-party setup consisting of at least **four participants**. A major advantage is the possible performance boost, since there can be more than one server that handles the computational effort, which is possibly very high, especially when using a key length of 2,048 bits or more. An obvious disadvantage is that more members need to be entrusted with private data. Even though data always remain encrypted, there are vulnerabilities nonetheless. For example, a corrupt administrator could try to track the amount of successful authentication attempts for each enrolled sample and use this information and their domain knowledge to match the samples to actual persons.

The multi-party setup allows every member to be in the setup more than once, which can be of interest for locally distributed systems. For example, a verifier appears multiple times and so, the database-holding participant, which implies that data can be saved either redundantly or distinctly. If, in a decentralized biometric access system, the servers keep their data distinct, every verifier has to keep on searching on the next server until every server has been checked or the data collection has been found. The case in which data are saved redundantly implies that there are as many participants possessing the whole data collection as there are servers. This does not only result in a performance boost. Each copy of the data collection adds a potential corrupt database holder, but makes it harder to keep track users. Furthermore, if a user has to be removed from the system, every trace has to be deleted, too. This is due to prevent reconstruction or information leakage, see also Section II.

A forensically secure deletion, see for instance [2], [3], becomes more complicated the more copies exist, especially if they are distributed on different servers. There can be multiple sensors in the system. It is obviously insecure, if all of them have access to the secret keys. Only if necessary security requirements are met and if the client is fully trusted, the access to the secret key can be granted. Actually, a sensor does not need the secret key to authenticate or enroll a user. Only when it comes to obtaining further information, for example the biometric sample itself as plain, the secret key is required.

Current approaches enable data mining techniques for user tracking. An adequate consideration of multi-participants is another open challenge for authentication. In the next section, we present our padding approach for an enhanced security similarity verification.

IV. Extending Secure Similarity Verification

There exist many biometric authentication systems, which use quite different biometric modalities. Another aspect in this domain is the quality of systems with respect to accuracy and security. To some extent, both properties rely on the trait itself. So, a system that uses only a small set of features with low quality is expected to have overlapping features for different users, which means an encryption of the same value with the same public key.

Due to the fact that systems often have more than one server and are using different key pairs, user tracking is not possible. Additionally, the order of users can be mixed within different systems. We introduce the padded biometrics approach, which allows user authentications in a multiple participant scenario with respect to privacy-preservation. Additionally, we present performance impacts and a brief security impact discussion.

A. The Padded Biometrics Approach

In Figure 4, we depict a scenario for user tracking with two database systems (DBS). We assume, an attacker has read access to both databases. The differences between both DBS are key pairs and user IDs. Assume, with some knowledge, the attacker identifies in DBS_1 User 1. The DBS uses an unsalted asymmetric encryption, which results for a given key and plain text value always in the same cipher value. Within DBS_1, the attacker finds the exact same value for another user (User 5). With the help of this knowledge, both users can be identified in DBS_2, see User 11 and User 31 in Figure 4. Due to the fact that the feature vectors are not shuffled, the attacker needs to identify a match between two users in DBS_2 with an overlap of the same two features.

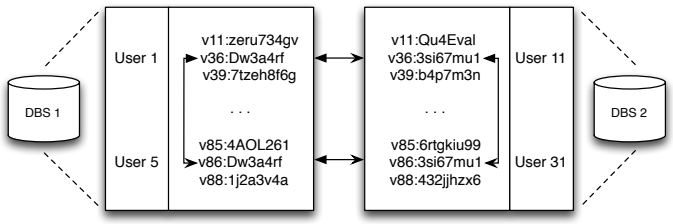


Figure 4. User Tracking in a Multiple DBS Setup, adapted from [27]

In practice, for a proper biometric trait with an appropriate resolution this scenario is implausible. As an example, we take the iris codes with 2,048 bit representation for the iris features; there exist theoretically more than 10^{74} different codes. However, the Euclidean vector space is very sparsely populated due to cluster of iris codes. Such clustering occurs in many biometric modalities. Therefore, our example, given in Figure 4, is a result from exact matches for different feature vectors. Correlations of biometric features are the main reason for such clusters. For instance, [28] examines different approaches in spatial domain iris data.

Daugman [29] identifies the iris phase code to be 0 or 1. This results in a Hamming distance with a very small variance. Daugman uses 249 different features and obtains $\mu = 0.499$ and $\sigma = 0.0317$. There exist several other analogous examples, e.g., in face recognition for the distribution of eyes, nose, and mouth that are quite similar for every person. We conclude that it is very likely that the data in the feature space are not equally distributed.

With these insights or domain knowledge, it is possible to link users or even track users as in our example in Figure 4. An

inclusion of the metadata of the database also enables further possibilities for an information gain, e.g., in the case that an index structure relates similar values, as the R-tree [30] or the Pyramid technique [31].

We propose a padding approach. This is comparable to salting [32]. In Figure 5, we show the idea. Every user receives a specific ID (UID). This ID is encrypted together with the template, e.g., by concatenating ID and biometric feature. This approach also allows including the feature index (FID) in the pad, which avoids intra-user overlapping.

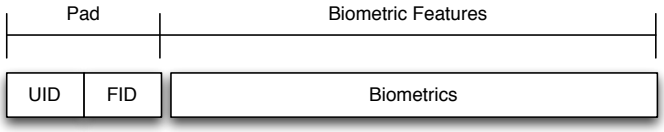


Figure 5. Lead-Pad for Biometric Features, adapted from [27]

The resulting value of a pad and a biometric feature has to be encrypted. A leading pad avoids any inter- and intra-user redundancies. At the same time, the possibility of the above described attack is close to zero. The padding, seen as a security layer, can be either maintained by the user or operated by an additional participant who has paddings and IDs.

This proposal comes at the cost that identification is expected to be more difficult. The pad shifts features semantically away from others. Therefore, the Euclidean measurements for similarity cannot be used, but the complete set of pads for each person has to be processed. We concentrate on performance of our proposed approach in the following.

B. Performance of the Padding Approach in the DBS

Index methods are widely used in DBMS to increase performance [33]. In relational databases, the B-tree [34] [35] and variants, such as the B+-tree, are used to achieve a logarithmic lookup performance. A similarity search using B+-trees results on average in a linear performance overhead additionally. Including a verifier, as proposed in an encrypted data domain, influences the processing time due to transportation effort. We discuss pros and cons in the following.

TABLE I. Performance in a Database System and compared to the Padding Approach

Query Type	DBS with B+-tree	Padding DBS
Exact Match	$O(log(n))$	$O(n)$
Similarity Search	$O(n)$	$O(n)$

Sorting and the use of metadata, which can improve query response times, should be avoided for security reasons. This requirement is in contrast to typically used index structures in relational data management systems. Therefore, the identification within the authentication process requires linear computational effort. Depending on the size and the application scenario, different metadata, such as gender, can be utilized to limit this effort. Note, if small subsets can be created from this metadata, it is necessary to separate these from biometrics. Alternatively, the padding approach can be applied to non-biometrics, too. In Table I, we summarize the computational efforts for a relational database and also for a database with encryption using our padding approach. Due to several other possible performance impacts, such as database size, feature size, thresholds, or key bit-length, we present in Section V a short evaluation study.

1) Implementation Issues: We propose to use a distance result from the verifier instead of a binary decision of acceptance or decline of an authentication attempt. Besides a reasonable attack scenario, where learning from accepted authentications and repeated authentication queries is possible in the later scenario, this risk can be reduced by disabling repeated authentication. In our approach, the quality of the similarity can be computed in an evaluation step. We apply the following formula:

$$d\left(X,Y\right) = \frac{\sum_{i=1}^{dim} |x_i - y_i|^a}{\tau^a \cdot dim} \tag{2}$$

with threshold τ, $a \geq 1$ as degree of freedom, and dim as dimensionality of the feature vector. These parameters are important for adjusting quality regarding sensor accuracy, error rates, and the biometric trait. The better the quality, the lower can be τ and the larger a.

We use a dictionary to maintain all pads for all enrolled users. The pads are delivered via a secure channel for each authentication process. The pads are concatenated before encryption. Due to the non-existence of relations to personal data, the pads can be generated randomly. The necessary step before enrollment or authentication is adding the pad. Note, it is not necessary to add the pad before the signal. Within an identification process, it is necessary to lookup the dictionary for the pad of a user. If outsourcing the dictionary to an external server, a processing time increase has to be respected.

In the following, we consider the three participant approach, for other system architectures from Section III. We measure the influence of computation time regarding all three involved participants. Note, if participants are embedded, as described in Section III, special security requirements have to be met.

The Three Participants scenario, as the default scheme considered in this paper, consists of a user, a verifier, and the DBS, which maintains the encrypted templates. Biometrics are taken by a sensor at user side. The verifier is responsible for authentication. Note, communication channels can be realized in different ways, such as insecure or with encryption. In the case, that only the user stores all pads to corresponding IDs, verifier and DBS do not need to be fully trusted. Hill-climbing should be avoided and therefore, a repeated authentication from single users has to be disabled. As a result, we can sum up that applying our approach to this scenario, only the user and partly the server gain information on the claimed identity. The ability to learn from the results can only be realized on user or verifier side. There is no plain information, due to encryption at user side within the complete process.

For the alternative scenarios discussed in Section III, we briefly present the alternatives in the following.

Two, respectively Two+ Participants: This system is comparable to the above described system. The difference is that the DBS is embedded at user side via a secure layer. We assume, users can never immediately access data by themselves, but only via strict protocols. Therefore, authentication is scheduled by the verifier. Whereas the embedding requires one participant less, secure embedding and sand-boxing of the DBS is necessary.

Three+ Participants (as an extension to the Three Participants scenario): Analogous to the Two+ Participants scenario, a secure layer is introduced. This connects user side and disguise. This requires either centralization or synchronization again. In this scenario, user side gains full information on IDs, but the DBS gains less information on their users. A trace of users is not possible due to random queries.

Four Participants: although the verifier learns in the three participants scenario the results from matching, this information is not important. However, to further decrease such a risk, a participant for disguising claimed identities is introduced. The function of this participant is to reduce the information gain for all participants. Therefore, the disguise blurs requested IDs by fake queries in undetermined intervals. It could also use a dictionary to reduce information gain at user side. However, the user has to learn a name or pseudo-identity to realize identity claiming. With a disguise participant neither the user nor the server can gain full information on claimed identities, but the disguise can learn this information. Verifier and user can learn from the results of the authentication.

In the following section, we show some impacts of our approach regarding security issues.

C. Security Impacts

Our experiments in this paper are separated into two groups: The experiments in the first group examine the security of the authentication system the second group focuses on performance issues. The objective of the security related experiments is to spot when and where data leakage can occur. Especially the changes to the original system proposed by Rane et al. in [11], [26] are of interest, because they affect the security of the system. An entire system analysis is not possible and a cryptanalysis would exceed the scope of this paper. Instead, a selection of conceivable attacks on the system is investigated. In these investigations the setup that causes additional risks for the confidentiality and privacy of the system are detailed.

Since many sensitive data sets are kept in DBS, this is a promising attack vector to gain information. A careful design, see Section IV-B1 and a proper security concept are mandatory. Implementation can cause vulnerabilities to the protocol that can lead to information leakage. There are some attacks, which do not immediately address the protocol. For instance, there are attacks on availability and the endpoint should be carefully considered. An attacker can try to take advantage of vulnerabilities that originated from poor system design. For example, a system designer decides to embed the verifier at user side, but does not meet all steps to guarantee confidentiality. If an unauthorized user is able to listen to the verifier, an information leakage occurs.

In the case the padding approach is implemented inappropriate, e.g., without secure separation from unauthorized users, and an attacker gains access to the pads, the confidentiality is at risk. With access to the pads and the encrypted signal, known-plain-text attacks [36] are possible.

Assume, the system design consists of a traditional DBS. This results in multiple instances of a dataset. Even though, all datasets are marked as deleted, it must be guaranteed on all DBS that there are no cached tables or backups available. So, every additional DBS requires a check that no information is left that can be used to recreate the data set. MySQL, for instance, only marks data with a certain bit, if data are deleted. The data are available in data slacks until they are overwritten. Furthermore, new data do not have the same length. If old data have not been overwritten by, for instance, NULL values, parts of old data can still remain. Accordingly, the data must be erased manually. Grebhahn [2] discusses this example in more detail.

The asymmetric Paillier cryptosystem as well as our padding approach are not information-theoretically secure compared to the symmetric approach, e.g., One-Time-Pad approach.

Thus, there are threats, such as the known-plain-text attack [36], leading to leakage of the biometric templates in the DBS. We introduce a padding approach to avoid opportunity of such attacks. Note, a secure dictionary is mandatory. The implementation of a system can enable various security vulnerabilities. These enable an attacker to gain trusted information. It is mandatory to implement a proper pseudonymization approach in combination with a secure dictionary.

The configuration of a system is presumably the most promising path for an attacker. The DBS amount and type of meta-information can be a threat to security. For instance, time stamps and logging information can be used to compromise security. An attacker can match users to datasets and therefore, trace users. So, system designers have to carefully consider meta-information. Additionally, backups play an important role. With access to both, DB and backup, an attacker subtracts users from backup and current state for user tracking.

Acceptance threshold and quality classes influence *false acceptance* and *false rejection rates*. The threshold decides on size of error patterns. There are many additional factors: level of information confidentiality, quality of the signals, access frequency, expectations regarding response times, and combined biometrics.

If the authentication protocol uses web communication, a denial of service attack (DoS) can disturb the protocol from functioning and harms availability. Even without using the web, there are other possible attacks that are not only taking advantage of communication. For instance, using malware to prevent participants from following the protocol is an imaginable attack on availability. Assuming that a biometric authentication scheme applies the Four Participants scenario,

a DoS attack on the disguise would prevent the system's functioning. It is possible to reduce the threat, but impossible to prevent it completely.

Endpoint security is crucial to provide confidentiality, especially if users have access to secret keys. Assuming the secret key is not as easily accessible, an attacker can try to read parts of communications. This includes plain and encrypted data such as pads. Assessing these data, follow-up attacks like known-plain or known cipher text attacks [36] are possible. For a restriction, basic security steps, including anti-virus software and firewalls, should be implemented.

V. Evaluation

In this section, we present evaluation results on performance for our approach. We focus on performance issues regarding our pad approach. Furthermore, we evaluate processing time as performance metric.

For our evaluation, we present experiments regarding different influence factors, such as enrolled users, key length, feature vector dimension, and thresholds. Firstly, we explain the evaluation setting. Secondly, we show results of our performance evaluation with respect to enrolled users, key length, feature dimensions, and threshold by studying with and without-padding approaches and encrypted versus non-encrypted scenarios.

A. Experimental Layout

For our evaluation, we use a MySQL database, version 5.5.27. We restrict our evaluation to a two table layout with index structures as follows:

- *Person*(Name, Security level, Department, ID)
- *Biometrics*(Feature, ID, BID).

Every enrolled person in the system has some attributes, i.e., a name, a security level, and a department. These attributes can be exchanged or extended by any property. In addition, every person has an ID to find a data tuple unambiguously. All properties like name, security level and department are encrypted with the public key. Biometrics are divided by the count of dimensions of the Euclidean vector. Every feature is identified by a biometric ID (BID), while biometrics are assigned to the corresponding person by an ID.

We make the following assumptions: The DBS is designed that it can be used for most common discrete biometric features. The resolution or the quality of the feature has no influence on the operative readiness of the biometric system itself. How accurate the resolution has to be is a question of acceptable error rates and needs to be adjusted by the corresponding use case. A forensic comparison of found biometrics on a crime scene, for example, needs to be well adjusted and requires a high quality of signals, while an attendance check must not be as accurate. Features are saved in feature vectors and have a minimum of at least one dimension and can have as many dimensions as needed. Everything that depends on the dimension of the feature vector grows corresponding to its size. For example, the codebooks are depending on the size of the feature vector.

B. Performance Evaluation

We perform all experiments on an AMD Phenom II X6 1055T Processor, an SSD, and 8GB RAM. In our evaluation, we focus on response time as crucial performance factor. We apply 10 replications per evaluation run for validity. We use artificial data that we $i.i.d.$ generated from Gaussian distribution. Note, there might be different parameters or measures. However, for simplicity, we exclude more complex influence parameters, such as skewness or correlation within our data. This does not simplifies our evaluations, but enables an easier identification of impacts.

First, we test for size of enrolled users. Note, for simplicity, the feature length is eleven dimensions, the key size is 64bit, and the threshold is set to three.

TABLE II. Performance for Users

Users	Database Processing Time per Users in ms	Identification in ms	Verification in ms
20	17	35	87
1,000	26	63	107
100,000	105	354	354

In Table II, we present arithmetic means for identification and verification for our padding approach. Our results indicate that the overall processing increases with a higher amount of enrolled users. This growth seems linear compared to the database processing time per users. With an increase of the database size, the processing time increases, too. For a sound comparison, we use a standardization of processing time per user. Memory management and thread scheduling or configuration and running the DBS cause this increase. Since verification only requires data of one person, the increase is not similar to identification. Due to B+-trees in MySQL, there is an increasing impact according to the size of enrolled users.

In Table III, we present results regarding key length. Note, we use 1,000 enrolled users in the DBS and a feature dimensionality of 11. As expected, an exponential growth with an increase of the key length is obvious. Due to our experimental setup (using one machine for all tasks), this growth might be influenced in our experimental setup. However, using a private key only increases the processing time in a small amount. A fast feedback is a user requirement for user acceptance of biometric authentication.

TABLE III. Performance for Key Length

Key Length	Identification in ms	Verification in ms
64	63	107
128	112	87
512	1001	400
1,024	6933	2188

We test different feature vector sizes (11, 69, 100, 250, and 2,048) and present the results in Table IV. Adding new features to the feature vectors requires more comparisons, which result in higher response times. Note, with an increase of the feature

TABLE IV. FEATURE DIMENSIONS PERFORMANCE

Feature Dimensions	Identification in ms	Verification in ms
11	63	56
69	239	81
100	354	321
250	693	571
2,048	1,065	860

vector the codebooks also increase. Due to this, the growth in smaller feature vectors can be explained.

As a last evaluation parameter, we vary the threshold from 3 to 1,000 and present our results in Table V. The threshold parameter is used for quality reasons, see also Section IV.

Compared to [11], increasing the threshold by 1 means that two additional comparisons have to be computed. Therefore, the increase is linear with the number of enrolled users. Signals with a higher fluctuation, which require a larger range of validity, require more processing time. This has to be examined for each application and evaluated regarding hardware, requirements, and accuracy.

Nevertheless, our experiment results show this linear relation in both settings, identification and verification. Note, the verification is slightly faster than the identification, which is comparable to the used feature dimensions.

TABLE V. PERFORMANCE FOR THRESHOLD

Threshold	Identification in ms	Verification in ms
3	125	99
5	199	104
10	216	114
100	280	208
1,000	1,572	1,311

As a concluding remark, we present our evaluation results regarding our approach compared to the approach presented in [11], which is implemented without a salting scheme. Note once again, salting increases privacy.

TABLE VI. PERFORMANCE FOR THE PADDING APPROACH

System Parameters	Padding Approach in ms	Without Pad in ms
1,000 users, 2,048 features, 64 bit	25,546	26,014
1,000 users, 11 features, 1,024 bit	28,033	27,197
100,000 users, 11 features, 64 bit	35,009	35,403

In Table VI, we show three different parameter scenarios exemplary. This table shows unexpected results. In the first and third experiment, the response times for the padding approach are slightly lower than without padding. This might be a result from caching and optimizations that take place in the experiments. Note, we conducted the experiment ten times to average execution overhead fluctuations. However, our results show that the influences of our approach are negligible.

In the last setting, we show differences between encrypted and unencrypted identification in Table VII. We use again a key length of 64bit and 11 feature dimensions. The threshold is set to 3. The results show the cost for encryption. Note, we only use a very small computation effort regarding encryption due to a very short key length. With an increase of the key length the difference for both scenarios increases dramatically.

TABLE VII. COMPARISON OF SECURE IDENTIFICATION

Enrolled Users	Encrypted Identification	Unencrypted Identification
20	35 ms	26 ms
1,000	63 ms	47 ms
100,000	354 ms	310 ms

Summarizing our evaluation, we state that a privacy-preserving encryption strategy is not only possible, but the processing overhead is acceptable. We provide a solution that is applicable to existing database systems. Furthermore, we show the impact on performance, which has to be considered in applying this approach.

VI. SUMMARY AND OUTLOOK

In this paper, we present an extension to the secure and similarity verification between homomorphically encrypted signals by Rane [11], [26]. Tracing users is possible in the original scenario. We present a padding approach, to overcome this challenge. We extend the original contribution to search on encrypted values and to use a padding concept. Furthermore, we develop a evaluation study of our conceptual design to evaluate our approach.

With the padding approach, an advanced search in an encrypted domain is possible. However, if repeated authentication attempts are possible, it is already possible to gain information regarding the template. One can avoid such template reproduction by disabling repeated authentications. Our approach improves data security. We name some security requirements for this purpose, but many attack scenarios are getting unlikely if information separation as well as signal processing in the encrypted domain are applied.

Processing times in our evaluation reveal that our padding approach comes at very low additional cost compared to [11]. This is an important aspect for user acceptance of such a system. Whereas the size of enrolled users has logarithmic impact on computational effort, the key length impacts with an exponential scheme. The dimensions of the feature vector have logarithmic influence as well and the threshold is linear in the computational effort. All these parameters do not drastically influence the system of Rane [11]. Due to simple operations, such as summation and amount computation, computational overhead is negligible. However, the concept of privacy-preserving authentication, discussed in this paper, has a strong influence on computational effort compared to plain-text biometric authentication systems.

In future work, our approach can be adapted for other domains that fulfill the same requirements on operations that need to be performed in the encrypted domain. We propose to semantically shift data to complicate unauthorized decryption

attempts, which makes user tracing via duplicate identification unlikely. Particularly, this becomes important, if the co-domain of the biometric feature is smaller than the co-domain of the key. The approach presented in [37] verifies users in the encrypted domain. It is imaginable that the extensions are of interest, too, for this approach, which bases on the homomorphic cryptosystem RSA.

ACKNOWLEDGMENTS

This work is partly based on the master thesis by Martin Leuckert [27]. We thank Martin Schäler for fruitful discussions on an earlier version of this paper. The work in this paper has been funded in part by the German Federal Ministry of Education and Research (BMBF) through the Research Program "DigiDak+ Sicherheits-Forschungskolleg Digitale Formspuren" under Contract No. FKZ: 13N10816 and 13N10818.

The work presented in this paper is based on the paper [1], which was presented at SECURWARE 2014. It contains original text from it with extensions to related work and architectural concepts.

REFERENCES

[1] J. Dittmann, V. Köppen, C. Krätzer, M. Leuckert, G. Saake, and C. Vielhauer, "Performance impacts in database privacy-preserving biometric authentication," in SECURWARE 2014: The Eighth International Conference on Emerging Security Information, Systems and Technologies, R. Falk and C. B. Westphall, Eds. IARA, 2014, pp. 111–117.

[2] A. Grebhahn, M. Schäler, and V. Köppen, "Secure deletion: Towards tailor-made privacy in database systems," in BTW-Workshops. Köllen-Verlag, 2013, pp. 99–113.

[3] A. Grebhahn, M. Schäler, V. Köppen, and G. Saake, "Privacy-aware multidimensional indexing," in BTW. Köllen-Verlag, 2013, pp. 133–147.

[4] C. Vielhauer, Biometric User Authentication for IT Security, ser. Advances in Information Security. Springer, 2006, no. 18.

[5] M. Schäler, A. Grebhahn, R. Schröter, S. Schulze, V. Köppen, and G. Saake, "QuEval: Beyond high-dimensional indexing à la carte," PVLDB, vol. 6, no. 14, 2013, pp. 1654–1665.

[6] F. Emekci, O. Sahin, D. Agrawal, and A. E. Abbadi, "Privacy preserving decision tree learning over multiple parties," DKE, vol. 63, no. 2, 2007, pp. 348 – 361.

[7] A. Inan, Y. Saygyn, E. Savas, A. Hintoglu, and A. Levi, "Privacy preserving clustering on horizontally partitioned data," in Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on, 2006, pp. 95–95.

[8] D. Shah and S. Zhong, "Two methods for privacy preserving data mining with malicious participants," Information Sciences, vol. 177, no. 23, 2007, pp. 5468–5483.

[9] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in EUROCRYPT, ser. Lecture Notes in Computer Science, J. Stern, Ed., vol. 1592. Springer, 1999, pp. 223–238.

[10] M. Schott, C. Vielhauer, and C. Krätzer, "Using different encryption schemes for secure deletion while supporting queries," in Datenbanksysteme für Business, Technologie und Web (BTW 2015) Workshopband, ser. Lecture Notes in Informatics, no. 242, Hamburg, 2015, pp. 37–45.

[11] S. Rane, W. Sun, and A. Vetro, "Secure similarity verification between homomorphically encrypted signals," US Patent US8 249 250 B2, Sep. 30, 2012.

[12] S. Kiltz, A. Lang, and J. Dittmann, "Taxonomy for computer security incidents," in Cyber Warfare and Cyber Terrorism. IGI Global, 2008, pp. 412–417.

[13] C. Vielhauer, J. Dittmann, and S. Katzenbeisser, "Design aspects of secure biometric systems and biometrics in the encrypted domain," in Security and Privacy in Biometrics, P. Campisi, Ed. Springer, 2013, pp. 25–43.

[14] S. Hildenbrand, D. Kossmann, T. Sanamrad, C. Binnig, F. Faerber, and J. Woehler, "Query processing on encrypted data in the cloud," Systems Group, Department of Computer Science, ETH Zurich, Tech. Rep., 2011.

[15] S. T. Peddinti and N. Saxena, "On the effectiveness of anonymizing networks for web search privacy," in Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, ser. ASIACCS '11. New York, NY, USA: ACM, 2011, pp. 483–489.

[16] D. Herrmann, C. Banse, and H. Federrath, "Behavior-based tracking: Exploiting characteristic patterns inDNS traffic," Computers & Security, vol. 39, Part A, no. 0, 2013, pp. 17 – 33, 27th {IFIP} International Information Security Conference.

[17] V. Köppen, M. Hildebrandt, and M. Schäler, "On performance optimization potentials regarding data classification in forensics," in Datenbanksysteme für Business, Technologie und Web (BTW 2015) Workshopband, ser. Lecture Notes in Informatics, no. 242, Hamburg, 2015, pp. 21–35.

[18] M. Schäler, S. Schulze, R. Merkel, G. Saake, and J. Dittmann, "Reliable provenance information for multimedia data using invertible fragile watermarks," in 28th British National Conference on Databases (BNCOD), ser. LNCS, vol. 7051. Springer, 2011, pp. 3–17.

[19] M. Schäler, "Minimal-invasive provenance integration into data-intensive systems," Ph.D. dissertation, Otto-von-Guericke-University, Magdeburg, Germany, DEC 2014.

[20] A. Juels and M. Wattenberg, "A fuzzy commitment scheme," in 6th ACM Conference on Computer and Communications Security. New York, NY, USA: ACM, 1999, pp. 28–36.

[21] A. Juels and M. Sudan, "A fuzzy vault scheme," Designs, Codes and Cryptography, vol. 38, no. 2, 2006, pp. 237–257.

[22] A. K. Jain, A. Ross, and U. Uludag, "Biometric template security: Challenges and solutions," in In Proceedings of European Signal Processing Conference, 2005.

[23] P. Stahlberg, G. Miklau, and B. N. Levine, "Threats to privacy in the forensic analysis of database systems," in Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD '07. New York, NY, USA: ACM, 2007, pp. 91–102.

[24] A. Grebhahn, D. Broneske, M. Schäler, R. Schröter, V. Köppen, and G. Saake, "Challenges in finding an appropriate multi-dimensional index structure with respect to specific use cases," in Proceedings of the 24th GI-Workshop "Grundlagen von Datenbanken 2012", I. Schmitt, S. Saretz, and M. Zierenberg, Eds. CEUR-WS, 2012, pp. 77–82, urn:nbn:de:0074-850-4.

[25] C. Gentry, "Computing arbitrary functions of encrypted data," Commun. ACM, vol. 53, no. 3, 2010, pp. 97–105.

[26] S. Rane, W. Sun, and A. Vetro, "Secure similarity verification between encrypted signals," US Patent US20 100 246 812 A1, Sep. 30, 2010.

[27] M. Leuckert, "Evaluation and extension of secure similarity verification in multi-computer scenarios to sesecure store and communicate biometric data," Master's thesis, Otto-von-Guericke University, 2013.

[28] M. Negin, T. A. Chmielewski, M. Salganicoff, T. A. Camus, U. M. C. von Seelen, P. L. Venetianer, and G. G. Zhang, "An iris biometric system for public and personal use," Computer, vol. 33, no. 2, 2000, pp. 70–75.

[29] J. Daugman, "How iris recognition works," IEEE Trans. on Circuits and Systems for Video Technology, vol. 14, no. 1, 2004, pp. 21–30.

[30] A. Guttman, "R-trees: A dynamic index structure for spatial searching," SIGMOD Rec., vol. 14, no. 2, 1984, pp. 47–57.

[31] S. Berchtold, C. Böhm, and H.-P. Kriegel, "The Pyramid-technique: Towards breaking the curse of dimensionality," SIGMOD Rec., vol. 27, no. 2, 1998, pp. 142–153.

[32] R. Morris and K. Thompson, "Password security: A case history," Commun. ACM, vol. 22, no. 11, Nov. 1979, pp. 594–597.

[33] V. Köppen, M. Schäler, and R. Schröter, "Toward variability management to tailor high dimensional index implementations," in RCIS. IEEE, 2014, pp. 452–457.

[34] R. Bayer and E. McCreight, "Organization and maintenance of large ordered indexes," Acta Informatica, vol. 1, 1972, pp. 173–189.

[35] D. Comer, "The Ubiquitous B-Tree," ACM Comput. Surv., vol. 11, no. 2, 1979, pp. 121–137.

[36] B. Schneier, Secrets & Lies: Digital Security in a Networked World. New York, NY, USA: John Wiley & Sons, Inc., 2000.

[37] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. V. Jawahar, "Efficient biometric verification in encrypted domain," in 3rd International Conference on Advances in Biometrics, 2009, pp. 899–908.

www.iariajournals.org

International Journal On Advances in Intelligent Systems
issn: 1942-2679

International Journal On Advances in Internet Technology
issn: 1942-2652

International Journal On Advances in Life Sciences
issn: 1942-2660

International Journal On Advances in Networks and Services
issn: 1942-2644

International Journal On Advances in Security
issn: 1942-2636

International Journal On Advances in Software
issn: 1942-2628

International Journal On Advances in Systems and Measurements
issn: 1942-261x

International Journal On Advances in Telecommunications
issn: 1942-2601