Gesellschaft für Informatik e.V. (GI)

publishes this series in order to make available to a broad public recent findings in informatics (i.e. computer science and information systems), to document conferences that are organized in cooperation with GI and to publish the annual GI Award dissertation.

Broken down into
• seminars
• proceedings
• dissertations
• thematics
current topics are dealt with from the vantage point of research and development, teaching and further training in theory and practice. The Editorial Committee uses an intensive review process in order to ensure high quality contributions.

The volumes are published in German or English.

Information: http://www.gi.de/service/publikationen/lni/

ISSN 1617-5468
ISBN 978-3-88579-650-3

This volume contains contributions of the 8th conference of the GI special interest group „Sicherheit, Schutz und Zuverlässigkeit“ that took place in Bonn (April 5-7, 2016). The main aspects of the conference were mobile security, malware analysis, privacy protection, IT security risk assessment, e-commerce, automotive security, Internet of Things security, and biometrics.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016

256

GI-Edition

Lecture Notes in Informatics

Michael Meier, Delphine Reinhardt, Steffen Wendzel (Hrsg.)

Sicherheit 2016

Sicherheit, Schutz und Zuverlässigkeit

Beiträge der 8. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)

5.–7. April 2016
Bonn

Proceedings

Michael Meier, Delphine Reinhardt, Steffen Wendzel (Hrsg.)

SICHERHEIT 2016

Sicherheit, Schutz und Zuverlässigkeit

Konferenzband der 8. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)

5.-7. April 2016
in Bonn

Gesellschaft für Informatik e.V. (GI)

Lecture Notes in Informatics (LNI) - Proceedings
Series of the Gesellschaft für Informatik (GI)

Volume P-256
ISBN 978-3-88579-650-3
ISSN 1617-5468

Volume Editors
Prof. Dr. Michael Meier
Universität Bonn / Fraunhofer FKIE
Friedrich-Ebert-Allee 144, D-53113 Bonn
E-Mail: mm@cs.uni-bonn.de
Jun.-Prof. Dr.-Ing. Delphine Reinhardt
Universität Bonn / Fraunhofer FKIE
Friedrich-Ebert-Allee 144, D-53113 Bonn
E-Mail: delphine.reinhardt@cs.uni-bonn.de
Dr. Steffen Wendzel
Fraunhofer FKIE
Friedrich-Ebert-Allee 144, D-53113 Bonn
E-Mail: steffen.wendzel@fkie.fraunhofer.de

printed by Köllen Druck+Verlag GmbH, Bonn

Vorwort

Die Fachtagung *Sicherheit, Schutz und Zuverlässigkeit* (SICHERHEIT) ist die regelmäßig stattfindende Tagung des Fachbereichs *Sicherheit – Schutz und Zuverlässigkeit* der Gesellschaft für Informatik e.V. (GI). Die Tagung deckt alle Aspekte der Sicherheit informationstechnischer Systeme ab und versucht eine Brücke zu bilden zwischen den Themen IT Security, Safety und Dependability.

Die achte Ausgabe der SICHERHEIT fand im April 2016 in Bonn statt und bot einem Publikum aus Forschung, Entwicklung und Anwendung ein Forum zur Diskussion von Herausforderungen, Trends, Techniken und neuesten wissenschaftlichen und industriellen Ergebnissen.

Erstmals fand auf der SICHERHEIT 2016 ein *Doktoranden-Forum* statt, das Doktoranden die Gelegenheit bot ihre Forschungs- und Dissertationsvorhaben zu Aspekten der Sicherheit informationstechnischer Systeme zu präsentieren, sich mit anderen Doktoranden sowie erfahrenen Wissenschaftlern auszutauschen und Kontakte über die eigene Universität hinaus zu knüpfen. Zudem bereicherten eine *Special Session* zum DFG Schwerpunktprogramm „Reliably Secure Software Systems“, ein *Practitioners Track*, der *International Workshop on Security, Privacy and Reliability of Smart Buildings* sowie das Finale des *Promotionspreises IT-Sicherheit 2015/2016* die Tagung.

Der vorliegende Tagungsband umfasst insgesamt 26 Beiträge. Traditionsgemäß durften Beiträge sowohl in Deutsch als auch in Englisch eingereicht werden. 15 Beiträge wurden aus 44 akademischen Einreichungen zur Haupttagung ausgewählt. Weiterhin wurden zwei Beiträge zum *Practitioners Track*, fünf Beiträge zum *Doktoranden-Forum* und vier Beiträge zum *International Workshop on Security, Privacy and Reliability of Smart Buildings* von den Programmkomitees ausgewählt und in den Tagungsband aufgenommen.

Neben den eingereichten Beiträgen und der *Special Session* konnten für die Tagung Marit Hansen (Landesdatenschutzbeauftragte Schleswig-Holstein) und Andreas Könen (Vizepräsident des Bundesamtes für Sicherheit in der Informationstechnik) für eingeladene Vorträge gewonnen werden.

Unser Dank gilt allen, die zum Gelingen der Tagung SICHERHEIT 2016 beigetragen haben. Allen voran sind hier die Autorinnen und Autoren, die Mitglieder der Programmkomitees sowie die externen Gutachter zu nennen. Unser besonderer Dank gilt dem gesamten lokalen Organisationsteam. Nicht zuletzt gilt unser Dank auch dem Leitungsgremium des GI-Fachbereichs Sicherheit – Schutz und Zuverlässigkeit, das uns tatkräftig bei der Organisation der Tagung unterstützt hat.

Michael Meier
Delphine Reinhardt
Steffen Wendzel

März 2016

Sponsoren

Wir danken den folgenden Unternehmen und Institutionen für die Unterstützung der Konferenz.

Huawei

Palo Alto Networks

Siemens

CAST e.V.

Genua

it.sec

Microsoft

ROHDE & SCHWARZ SIT

XING Events

Tagungsleitung der Sicherheit 2016

Hannes Federrath, Universität Hamburg
Stefan Katzenbeisser, Technische Universität Darmstadt
Michael Meier, Universität Bonn und Fraunhofer FKIE (General Chair)
Steffen Wendzel, Fraunhofer FKIE (Program Co-Chair)

Lokale Organisation

Saffija Kasem-Madani	Christian Kollee	Arnold Sykosch
Jaspreet Kaur	Jamila Krämer	Jernej Tonejc
Alexandra Kobekova	Hendrik Rommerskirchen	Matthias Wübbeling

Programmkomitee

Frederik Armknecht, Universität Mannheim
Bernhard Beckert, Karlsruher Institut für Technologie
Thomas Biege, SUSE GmbH
Rainer Böhme, Universität Innsbruck
Jens Braband, Siemens und Technische Universität Braunschweig
Arslan Brömme, Vattenfall GmbH
Christoph Busch, Hochschule Darmstadt
Peter Dencker, Hochschule Karlsruhe
Jana Dittmann, Universität Magdeburg
Wolfgang Ehrenberger, Hochschule Fulda
Bernhard Fechner, FernUniversität Hagen
Hannes Federrath, Universität Hamburg
Ulrich Flegel, Infineon
Felix Freiling, Universität Erlangen-Nürnberg
Karl-Erwin Grosspietsch, Euromicro
Hannes Grunert, Universität Rostock
Andreas Heinemann, Hochschule Darmstadt und CASED
Maritta Heisel, Universität Duisburg-Essen
Jörg Helbach, E&P Service GmbH
Eckehard Hermann, Fachhochschule Hagenberg
Dominik Herrmann, Universität Siegen
Thorsten Holz, Universität Bochum
Detlef Hühnlein, ecsec GmbH
Dieter Hutter, DFKI und Universität Bremen
Jan Jürjens, Technische Universität Dortmund
Stefan Katzenbeisser, Technische Universität Darmstadt
Hubert B. Keller, Karlsruher Institut für Technologie
Jörg Keller, FernUniversität Hagen
Dogan Kesdogan, Universität Regensburg
Herbert Klenk, Airbus
Dirk Koschützki, Hochschule Furtwangen

Kerstin Lemke-Rust, Hochschule Bonn-Rhein-Sieg
Luigi Lo Iacono, Technische Hochschule Köln
Heiko Mantel, Technische Universität Darmstadt
Michael Meier, Universität Bonn und Fraunhofer FKIE
Matthias Meyer, GDATA
Ulrike Meyer, Rheinisch-Westfälische Technische Hochschule Aachen
Holger Morgenstern, Hochschule Albstadt-Sigmaringen
Isabel Münch, BSI
Michael Nüsken, b-it Bonn
Frank Ortmeier, Universität Magdeburg
Sebastian Pape, Goethe-Universität Frankfurt
Andreas Peter, Technische Universität Twente
Erhard Plödereder, Universität Stuttgart
Hans Pongratz, Technische Universität München
Joachim Posegga, Universität Passau
Alexander Pretschner, Technische Universität München
Kai Rannenberg, Goethe-Universität Frankfurt
Peer Reymann, ITQS GmbH
Konrad Rieck, Universität Göttingen
Volker Roth, Freie Universität Berlin
Heiko Roßnagel, Fraunhofer IAO
Francesca Saglietti, Universität Erlangen-Nürnberg
Peter Schartner, Universität Klagenfurt
Sebastian Schmerl, Computacenter
Karin Schuler, Expertin für Datenschutz und IT-Sicherheit
Torsten Schütze, ROHDE & SCHWARZ SIT GmbH
Jörg Schwenk, Universität Bochum
Claus Stark, Citigroup
Hermann Strack, Hochschule Harz
Thorsten Strufe, Technische Universität Dresden
Melanie Volkamer, Technische Universität Darmstadt und CASED
Christian Weber, Ostfalia - Hochschule für angewandte Wissenschaften
Edgar Weippl, SBA Research
Steffen Wendzel, Fraunhofer FKIE
Ernst O. Wilhelm, GFT Technologies AG
Bernhard C. Witt, it.sec GmbH & Co. KG

Zusätzliche Gutachter

Svetlana Abramova
Daniel Arp
Thomas Bauereiß
Martin Beck
Arne Bilzhause
Toralf Engelke
Marco Ghiglieri
Daniel Grahl
Simon Greiner
Steffen Herterich
Stefanie Jasser
Vladimir Klebanov
Stefan Köpsell
Sebastian Kurowski
Stefan Laube
Jens Lindemann
Sebastian Luhn
Malte Möser
Henrich C. Poehls
Sebastian Reif
Christian Roth
Thomas Santen
David Schneider
Alexander Senier
Ralf C. Staudemeyer
Alexandra Weber
Alexander Weigl
Ephraim Zimmer

Tagungsleitung des Doktorandenforums „Sicherheit 2016“

Delphine Reinhardt, Universität Bonn und Fraunhofer FKIE (Chair)

Programmkomitee

Frederik Armknecht, Universität Mannheim
Rainer Böhme, Universität Innsbruck
Hannes Federrath, Universität Hamburg
Felix Freiling, Friedrich-Alexander-Universität Erlangen-Nürnberg
Thorsten Holz, Ruhr-Universität Bochum
Stefan Katzenbeisser, Technische Universität Darmstadt
Michael Meier, Universität Bonn und Fraunhofer FKIE
Ulrike Meyer, Rheinisch-Westfälische Technische Hochschule Aachen
Joachim Posegga, Universität Passau
Alexander Pretschner, Technische Universität München
Kai Rannenberg, Goethe-Universität Frankfurt
Konrad Rieck, Georg-August-Universität in Göttingen
Jörg Schwenk, Ruhr-Universität Bochum
Matthew Smith, Universität Bonn und Fraunhofer FKIE

Tagungsleitung des International Workshop on Security, Privacy and Reliability of Smart Buildings

Steffen Wendzel, Fraunhofer FKIE (Chair)
Jörg Keller, FernUniversität Hagen (Co-Chair)
Jernej Tonejc, Fraunhofer FKIE (Program Co-Chair)
Jaspreet Kaur, Fraunhofer FKIE (Website)

Programmkomitee

François-Xavier Aguessy, Thales
Bernhard Fechner, FernUniversität in Hagen
Christian Herdin, Hochschule Augsburg
Saffija Kasem-Madani, Universität Bonn
Wolfgang Kastner, Technische Universität Wien
Jaspreet Kaur, Fraunhofer FKIE
Jörg Keller, FernUniversität in Hagen
Wojciech Mazurczyk, Technische Universität Warschau
Nasim Mohammadi, York University
Delphine Reinhardt, Universität Bonn und Fraunhofer FKIE
Arnold Sykosch, Universität Bonn und Fraunhofer FKIE
Jernej Tonejc, Fraunhofer FKIE
Steffen Wendzel, Fraunhofer FKIE
Matthias Wübbeling, Universität Bonn und Fraunhofer FKIE
Tao Yue, University of Oslo
Sebastian Zander, Murdoch University

Zusätzliche Gutachter

Phu H. Nguyen Markus Wurzenberger Ma Tao

Inhaltsverzeichnis

Authentifikation und Biometrie

E-Commerce & Sicherheitsmanagement

Practitioners Track

Doktoranden-Forum "Sicherheit 2016"

Why 2 times 2 ain't necessarily 4 – at least not in IT security risk assessment

Jens Braband[1]

Abstract: Recently, a novel approach towards semi-quantitative IT security risk assessment has been proposed in the draft IEC 62443-3-2. This approach is analyzed from several different angles, e.g. embedding into the overall standard series, semantic and methodological aspects. As a result, several systematic flaws in the approach are exposed. As a way forward, an alternative approach is proposed which blends together semi-quantitative risk assessment as well as threat and risk analysis.

Keywords: IT security, risk analysis, semi-quantitative methods, risk matrix, IEC 62443.

1 Introduction

IEC 62443 is a new series of IT security standards for industrial control systems (ICS), which is currently being elaborated jointly by ISA and IEC. Recently, the first draft for Part 3-2, which deals with IT security risk assessment, has been circulated for comments [IEC3-2]. It contains a novel approach towards semi-quantitative IT security risk assessment, which will be discussed in this paper.

The paper is organized as follows: first, a short introduction to the general concepts of IEC 62443 is given, so that the new approach can be understood in its context. Then, this approach is analyzed formally and technically, in particular against good practices for semi-quantitative risk assessment in other sectors. By this analysis, some general flaws of the approach are revealed so that finally a discussion of how to possibly overcome these weaknesses is necessary.

2 Basic concepts of IEC 62443

2.1 Scope

A total of 12 standards or technical specifications are planned in the IEC 62443 series of standards that cover the topic of IT security for automation and control systems for industrial installations entirely and independently. This series of standards adds the topic of IT security to IEC 61508 which is the generic safety standard for programmable

[1] Siemens AG, Mobility Division, Ackerstrasse 22, 38126 Braunschweig, jens.braband@siemens.com

control systems. Up to now though, IEC 61508 and IEC 62443 have only been loosely linked.

IEC 62443 addresses four different aspects or levels of IT security:

- general aspects such as concepts, terminology and metrics: IEC 62443-1-x
- IT security management: IEC 62443-2-x
- system level: IEC 62443-3-x
- component level: IEC 62443-4-x

Today, however, the parts of IEC 62443 are still at different draft stages. Only a small number of parts such as IEC 62443-3-3 have already been issued as an international standard (IS) or a technical specification (TS). Due to the novelty of the IEC 62443 series in this section, the essential concepts of IEC 62443 will be explained briefly so as to improve the understanding of this paper.

2.2 System definition

The system and its architecture are divided into zones and conduits. The same IT security requirements apply within each zone. Every object, e.g. hardware, software or operator (e.g. administrator), shall be assigned to precisely one zone and all connections of a zone shall be identified. A zone can be defined both logically and physically. This approach matches the previous approach for railway signaling systems very well, as has been used as a basis in numerous applications [11]. Figure 1 shows a simple application of the concept, the connection of two safety zones (sharing the same security requirements) by a virtual private network (VPN) connection as the conduit.

The conduit would consist of the gateways at its borders and the connection in-between whatever the actual network would look like. Strictly speaking, management itself would be a zone with conduits connecting it with the gateways.

This example may serve as a blueprint for the connection of zones with similar IT security requirements. If zones with different IT security requirements are to be connected, different types of conduits, e.g. one-way connections or filters, have to be applied.

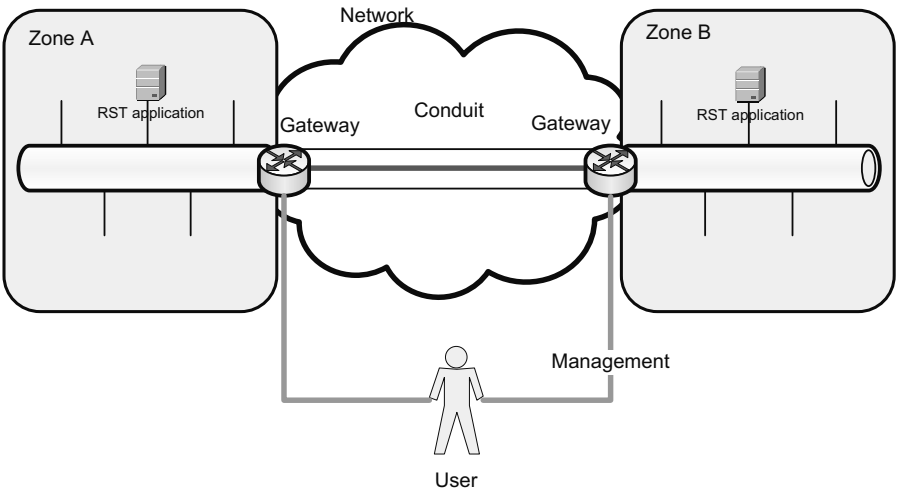


Fig. 1: Zone and conduit architecture example

2.3 IT security requirements

In IEC 62443, the IT security requirements are grouped into seven foundational requirements (FR):

1. identification and authentication control (IAC)
2. use control (UC)
3. system integrity (SI)
4. data confidentiality (DC)
5. restricted data flow (RDF)
6. timely response to events (TRE)
7. resource availability (RA)

Normally, only the issues of integrity, availability and data confidentiality are considered in IT security. However, the fundamental requirements IAC, UC, SI and TRE can be mapped to integrity, RA to availability, and DC and RDF to confidentiality. Instead of defining a seven-level evaluation assurance level (EAL) as in the Common Criteria, which is to be applied with regard to the IT security requirements, a four-stage IT security requirement level is defined. A possible explanation might be that also most safety standards define four levels. However, it would lead to quite demanding and sometimes unnecessary requirements if the levels were the same for each of the foundational requirements. For example, confidentiality often plays a minor role for safety systems and the encryption of all data might lead to complications in the testing or

maintenance of safety systems. Hence, different levels may be assigned for each of the seven foundational requirements. The SL values for all seven basic areas are then combined into a vector, called the SL vector. Note that this theoretically leads to 16,384 possible different SLs (only partially ordered).

The SLs are defined generically in relation to the attacker type against whom they are to offer protection:

SL 1 Protection against casual or coincidental violation

SL 2 Protection against intentional violation using simple means with few resources, generic skills and a low degree of motivation

SL 3 Protection against intentional violation using sophisticated means with moderate resources, IACS-specific skills and a moderate degree of motivation

SL 4 Protection against intentional violation using sophisticated means with extended resources, IACS-specific skills and a high degree of motivation

Sometimes, a SL 0 (No protection) is also defined, but, as we argue below, at least for safety-related systems this is not an option and so we do not discuss SL 0 further in this paper.

For one zone, for example, (4, 2, 3, 1, 2, 3, 2) could be defined as an SL vector. Once its vector is defined, IEC 62443-3-3 [IEC3-3] gives a complete catalog of standardized IT security requirements for the object under consideration, e.g. for a zone.

It is necessary to take into account the fact that IEC 62443 defines different types of SL vectors:

- The target SL (SL-T) is the SL vector that results as a requirement from the IT security risk analysis.
- Achieved SL (SL-A) is the SL vector which is actually achieved in the implementation when all the particular conditions in the specific system are taken into account.
- SL capability (SL-C) is the maximum SL vector that the components or the system can reach if configured or integrated correctly, independent of the framework conditions in the specific system.

3 New IT security risk assessment approach

Currently, there exists no agreed approach on how to derive a SL from a threat and risk Analysis (TRA). So [IEC3-2] came up with a new approach starting with a sample risk matrix as shown in Table 1. It looks like a common approach to determine the risk R for

a particular threat from the parameters likelihood L and impact I by

$$R = L \cdot I \tag{1}$$

		Likelihood				
		1 Remote	2 Unlikely	3 Possible	4 Likely	5 Certain
Impact	1 Trivial	1	2	3	4	5
	2 Minor	2	4	6	8	10
	3 Moderate	3	6	9	12	15
	4 Major	4	8	12	16	20
	5 Critical	5	10	15	20	25

Tab. 1: Sample risk matrix

Then, 4 is stipulated as a tolerable risk (without any justification) and any identified risk value R is then divided by 4, giving the so-called cyber security risk reduction factor (CRRF)

$$\text{CRRF} = \frac{R}{4} \tag{2}$$

and finally the target SL is derived by

$$\text{SL-T} = \min\left\{4, \left\lfloor \text{CRRF} - \frac{1}{4} \right\rfloor\right\} \tag{3}$$

A formula (3) simply states that a SL-T must not be larger than 4 and that it is more or less given by the integer part of the CRRF with a small correction of ¼. In order to understand it better, let us look at some interesting examples. For R=16, the CRRF is 4, which by (3) leads to SL-T=3. For R=17, it would lead to SL-T=4. Interestingly, both risks belong to the highest risk category in Table 1. Also, other border cases are interesting, e.g. risks labeled 6, 7 and 8 lead to SL-T=1, while 9 and 10 would result in SL-T=2. While all low-level risks should normally be acceptable, risks with 1, 2, 3, and 4 lead to SL-T=0, while 5 leads to SL-T=1. These initial observations are just an appetizer and an invitation for a more thorough analysis.

4 Analysis of the new approach

4.1 Embedding in IEC 62443

It must be clearly stated that Table 1 and (3) are only designated as examples by [IEC3-2]. But it is also clear from the process description that the example is at least meant as a blueprint. The overall process consists of the following steps

1. Identify threats
2. Identify vulnerabilities
3. Determine consequence and impact
4. Determine unmitigated likelihood
5. Calculate unmitigated cyber-security risk
6. Determine security level target
7. Identify and evaluate existing countermeasures
8. Re-evaluate likelihood and impact
9. Calculate residual risk
10. Compare residual risk with tolerable risk
11. Apply additional cyber-security countermeasures
12. Document and communicate results

As explained above, IEC 62443 derives fundamental requirements in seven different groups for zones and conduits of a particular IT security architecture, e.g. that of Figure 1. So the result should be a seven-dimensional SL-T vector instead of a scalar value given by (3). But the process description does not give any hint of how to derive the SL-T vector of a zone or conduit from the risk assessment of a threat-vulnerability combination. No explanation is given about how the concept is broken down to the foundational requirements. It may formally be argued that the authors assume that all components of the SL-T vector equal the scalar value derived by (3), but this would, in most cases, lead to very demanding requirements, e.g. for most ICS applications confidentiality is less important than integrity or availability and so the DC foundational requirement can be much weaker than that for SI or RA.

Also, at least for safety-related systems, SL-T=0 does not really make sense as protection against casual or coincidental violation should be provided in any case. It is hard to imagine a system which should not be protected against such threats. For safety-related systems, it is necessary to prevent human errors or foreseeable misuse in any case.

Additionally, there is a difference in the definition of SL between the proposal in IEC 62443-3-2 and the other parts of the standards. By applying formulae (2) and (3), the SL-T is equivalent to a risk reduction, while in the other parts, e.g. 62443-3-3, the SL-T is defined with respect to the attacker type against whom they are to offer protection. The relationship between risk reduction and the type of attacker is not explained, so it is questionable whether the approach fits to other parts of the standard.

4.2 Semantics

The input scales for parameters L and I are ordinal, so we know only the ordering of values 1<2<3<4<5, but have no knowledge about their further relations. For example, we do not know if an impact of 3 is five times more severe than that of 2. We could also re-label the categories to A; B, C, D, E [WP].

To make this more tangible, in programming languages such as Pascal or C, such ordinal types could be declared as

```
type
  impact = (trivial, minor, moderate, major, critical);
  likelihood = (remote, unlikely, possible, likely,
  certain);
```

Semantically, only certain operations such as predecessor, successor, ordinal number, greater than, etc., are defined for ordinal data types, but certainly not multiplication or division, which are simply undefined for ordinal data.

What is suggested by Table 1 is that the ordinal data such as "minor" is equated numerically with their order values in their type definition, e.g. `Ord(minor)` which equals 2. These order values are then treated as rational numbers and used without further explanation.

To make this argument clearer, assume that we would have labeled Table 1 with letters instead of numbers. What would $B \cdot C$ mean? Or how would the cyber-security risk reduction factor $B \cdot C/4$ be interpreted? And why should the values be multiplied and not be added?

4.3 Semi-quantitative risk assessment

Risk matrices like the example in Table 1 are often used in so-called semi-quantitative risk assessments such as the risk priority number (RPN) in failure modes effects and criticality assessment (FMECA). For this purpose, the classes are enriched by numerical values interpreted either as mean values for the class or as intervals.

It is well known that such approaches may have systematic flaws [Bow03], but approaches for improvement are also known [Bra03]. Here, we want to focus on the problem of multiplication only. For this discussion, we label the combination of the input parameters as the criticality of the scenario. Thus, the sample matrix in Table 1 contains the criticality numbers. The basic requirements for such approaches are:

1. If two scenarios bear a similar risk, then they should have the same criticality
2. If two scenarios are assigned to the same criticality, then they should represent similar risks

However, simple but realistic scenarios show that multiplication is not an appropriate operator with respect to these requirements. Figure 2 shows a typical example with numerical values as can also be found in standards. It can clearly be seen that both above requirements are not fulfilled as, for example, there could be scenarios that have the same criticality but the corresponding risks differ by a factor of almost one thousand. It has been shown [Bra03] that this effect is systematic and is similar for all examples. For this reason, appropriate caveats have nowadays been included in standards [IEC812].

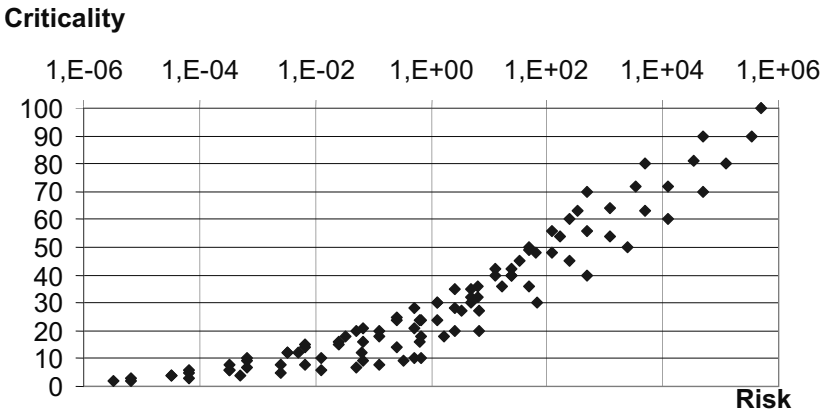


Fig. 2: Relationship between risk and criticality in RPN

So, even if the categories were defined semi-quantitatively, the basic approach as in (1) would still be flawed.

5 Way forward

We can summarize the analysis so far that the approach proposed by [IEC3-2] has several systematic flaws which cannot be easily overcome. In particular, the question of calculating IT security-related risks is very complex and should be avoided [BS15]. However, the use of risk matrices in IT security is so widely used in TRA that it should be kept, but it should be properly used with the definition of SL in IEC 62443.

We start from the following assumptions (without further justification):

- There exists an agreed risk matrix.
- The goal is to derive SLs which are defined by the type of attacker and the measures defined by IEC 62443.

For the sake of simplicity, we assume the same sample risk matrix as shown in Table 1 (but we do not use the criticalities). The precise form of the matrix is not important, however there should be a clear procedure which would be followed based on the color code of the results.

In a TRA, we would assess all possible threat scenarios and classify them according to their risk. The following example shows the result for three scenarios X, Y and Z. In this assessment, we have assumed that, for safety systems, we should always fulfill SL 1 =(1,1,1,1,1,1,1). This means that, in the TRA, we have assumed that all requirements related to this SL from [IEC3-3] are fulfilled, meaning that appropriate countermeasures corresponding to this SL have been implemented.

The result shows that we are OK for scenario X, but should improve for scenarios Y and Z. We now iteratively look at the assessments of the scenarios and look qualitatively for the features that should be improved from an IT security point of view. Assume in scenario Y we have a problem with authentication and user rights management. So we could increase the SL for the FR IAC and UC to (2,2,1,1,1,1,1). Let us assume this would be sufficient and we can move this scenario to the green field. But this could also have side effects on the other scenarios. Hence, we would have to re-evaluate.

		Likelihood				
		Remote	Unlikely	Possible	Likely	Certain
Impact	Trivial					
	Minor		X			
	Moderate				Z	
	Major		Y			
	Critical					

Tab. 2: Qualitative sample risk matrix

Assume in Z we have a problem with integrity, so we might also increase the SL for the FR SI to (2,2,2,1,1,1,1). If this is not sufficient, we would have to try (2,2,3,1,1,1,1) in order to make all risks acceptable after re-evaluation.

So we could use the TRA to iteratively find the SL for all zones and conduits. The TRA can be repeated at different stages in the IT security lifecycle, e.g. to determine SL-T, SL-C or SL-A.

Alternatively, we can also start to define SL by the type of attacker with another starting point, say initially SL is equal to (3,3,3,1,1,3,1), because we assumed that this level is

appropriate for the foundational requirements which we have selected. Then, we would start the TRA as a check and should arrive at the same result regarding risk tolerability as before. However, as we have started with more demanding requirements, as a side effect we may also have reduced the risk associated with other scenarios, e.g. X.

6 Summary and conclusions

Our analysis of the approach proposed in the draft IEC 62443-3-2 has shown the following systematic flaws:

- The tolerable risk is not justified.
- The calculation of ratios used for the ordinal data is not defined.
- SL is a seven-dimensional vector, but a scalar value is derived.
- The approach contradicts the definition of SL by the type of attacker in other parts of the standard.

By way of a summary, the approach is not only unjustified, but also dangerous because the user does not need to think qualitatively about IT security; the focus is clearly on pseudo-quantification pretending unjustified accuracy.

As a way forward, an alternative approach is proposed which blends together semi-quantitative risk assessment as well as threat and risk analysis.

References

[Bow03] Bowles, J.: An Assessment of RPN Prioritization in a Failure Modes, Effects and Criticality Analysis. Proceedings of RAMS 2003, Tampa, Florida, 2003

[Bra03] Braband, J.: Improving the Risk Priority Number Concept, Journal of System Safety, 02/2003, 21-24

[BS15] Braband, J., Schäbe, H.: Probability and Security – Pitfalls and Chances: in Proc. Advances in Risk and Reliability Technology Symposium 2015, Loughborough, 2015

[IEC3-2] IEC 62443-3-2: Security for industrial automation and control systems – Part 3-2: Security risk assessment and system design, draft for comments, August 2015

[IEC3-3] IEC 62443-3-3: Security for industrial automation and control systems – Part 3-3: System security requirements and security levels, international standard, 2015

[IEC812] IEC 60812: Analysis techniques for system reliability – Procedure for failure mode and effects analysis (FMEA), international standard, 2006

[WP] Wikipedia: Ordinal Scale, https://en.wikipedia.org/wiki/Level_of_measurement#Ordinal_scale, last accessed on 2015-11-25

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 11

Sicherheitsanalyse mit JOANA

Jürgen Graf, Martin Hecker, Martin Mohr, Gregor Snelting[1]

Abstract: JOANA ist ein System zur Software-Sicherheitsanalyse, das bis zu 100kLOC volles Java mit Threads analysieren kann. JOANA basiert auf modernen Verfahren zur Programmanalyse und erzeugt deshalb wenig Fehlalarme. JOANA enthält einen "iRLSOD" Algorithmus, der probabilistische Nichtinterferenz garantiert. JOANA ist Open Source, braucht wenige Annotationen und lässt sich leicht bedienen. Der Artikel gibt einen Überblick über JOANA, erklärt das neue iRLSOD Verfahren, und präsentiert eine Fallstudie.[2]

Keywords: Software-Sicherheit, Programmanalyse, Nichtinterferenz

1 Übersicht

Klassische Techniken der Software-Sicherheit, wie etwa Zertifikate, analysieren nicht das tatsächliche Verhalten von Programmen, und können deshalb keine wirklichen Garantien über Vertraulichkeit und/oder Integrität einer (heruntergeladenen) Software machen. *Informationsflusskontrolle* (IFC) ist eine neue Sicherheitstechnik, die international intensiv untersucht wird, um klassische IT-Sicherheit zu ergänzen. IFC analysiert den Quelltext oder Bytecode einer Software, und findet entweder alle Lecks, oder kann eine echte Garantie geben, dass die Software Vertraulichkeit und Integrität respektiert.[3] Dabei bedeutet ein Vertraulichkeits-Leck, dass geheime Daten das öffentlich sichtbare Verhalten der Software beeinflussen, und ein Integritäts-Leck, dass kritische Berechnungen von aussen manipuliert werden können. Inzwischen wurden verschiedene praktisch einsetzbare IFC-Tools entwickelt, die auf verschiedenen Sicherheitskriterien und Analyseverfahren basieren, und sich deshalb in Sprachumfang ("volles Java"), Präzision ("keine Fehlalarme"), Skalierbarkeit ("große Programme"), Benutzbarkeit ("wenig Aufwand"), Korrektheit ("garantiert keine Lecks") und anderen Aspekten unterscheiden.

Das IFC-Tool JOANA wurde am KIT entwickelt und ist öffentlich verfügbar (siehe `joana.ipd.kit.edu`). JOANA analysiert Java Bytecode oder Android (Dalvik) Bytecode. Für alle Ein- und Ausgaben muss der Prüfingenieur angeben, ob sie geheim ("high") oder öffentlich ("low") sind.[4] JOANA kann volles Java mit beliebigen Threads behandeln (nur bei Reflection gibt es gewisse Einschränkungen). JOANA skaliert bis ca. 100kLOC und bietet – dank moderner kontext-, fluss-, und objekt-sensitiver Analysealgorithmen – hohe Präzision und wenig Fehlalarme. In nebenläufigen Programmen werden auch die tückischen

[1] KIT, Fakultät für Informatik, Am Fasanengarten 5, 76131 Karlsruhe, graf@kit.edu

[2] Abschnitt 1-3 des vorliegenden Artikels erschienen in englischer Sprache in: Checking probabilistic noninterference using JOANA. it – Information Technology 56(6), S. 280–287, 2014.

[3] IFC analysiert *nur* den Code einer Applikation. IFC analysiert *nicht* Hardware, Betriebssystem, Internet-Pakete etc. Deshalb muss IFC zusammen mit herkömmlichen Sicherheitstechniken verwendet werden.

[4] JOANA kann beliebig komplexe Verbände von Sicherheitsstufen verwenden, nicht nur *high* und *low*.

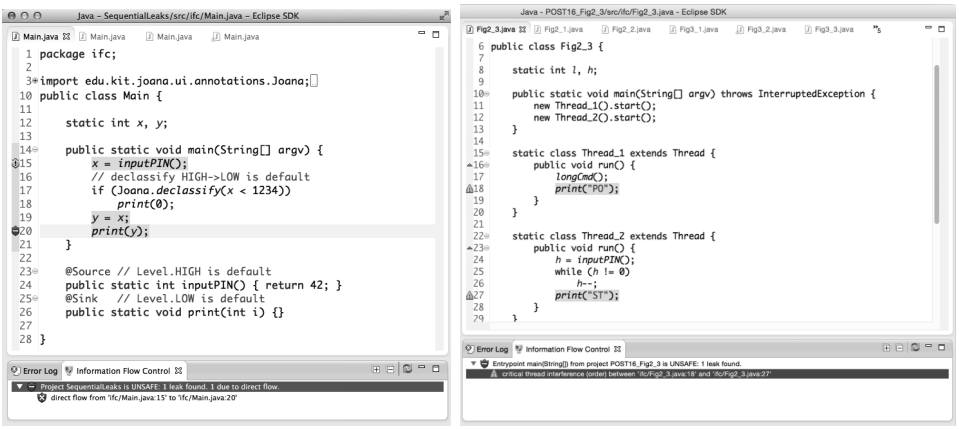


Abb. 1: JOANA GUI. Links: Beispiel mit Klassifikation, Deklassifikation, und illegalem Fluss. Rechts: Beispiel mit probabilistischem Leck.

probabilistischen Lecks entdeckt. Der JOANA Analysealgorithmus für sequentielle Programme wurde mit dem Maschinenbeweiser Isabelle verifiziert. JOANA wurde in diversen Fallstudien, u.a. Analyse eines E-Voting Prototyps angewendet.

Damit ist JOANA eines der wenigen international verfügbaren IFC-Tools, die reale Programme mit beliebigen Threads analysieren können. In diesem Beitrag sollen neue Ergebnisse im JOANA Projekt beschrieben werden, insbesondere das neue iRLSOD-Kriterium nebst Anwendungen. Die technischen Grundlagen sind im Detail in [HS09, GS15, WLS09, BGH$^+$16] beschrieben, spezifische Anwendungen u.a. in [KTG12, KSTG14, MGH15, GHMS15].

2 JOANA Anwendung

Abbildung 1 zeigt das JOANA GUI. Links ist im Quelltextfenster der vollständige Java-Code für Beispiel (1) aus Abbildung 2 zu sehen. Sicherheitsstufen für Ein- und Ausgaben sind als Annotationen hinzugefügt, ebenso wie eine Deklassifikation für `x` im `IF`. Die IFC-Analyse markiert illegale Flüsse im Quelltext. Im Beispiel wird der illegale Fluss von der geheimen `inputPIN` zum `print(y)` markiert, während der illegale Fluss von `inputPIN` zu `print(0)` durch die Deklassifikation "künstlich erlaubt" wird. Weitere Details werden auf Wunsch angezeigt. Abbildung 1 rechts zeigt die Analyse des probabilistischen Lecks analog Beispiel (3) in Abbildung 2, das weiter unten erklärt wird. JOANA markiert die beiden Anweisungen, deren Ausführungsreihenfolge von geheimen Daten abhängen kann und so einem Angreifer einen Ansatzpunkt bieten.

JOANA bietet verschiedene Optionen für die Präzision der Programmanalyse (z.B. objektsensitive Points-to Analyse, zeit-sensitives Slicing), und sogar einen Autotuning-Mechanismus zum Finden einer Konfiguration, die keine Fehlalarme auslöst [Gra16]. JOANA verwendet das IBM Analyseframework WALA als Front End. JOANA konnte Sicherheits-

(1)

```
1  void main():
2    x = inputPIN();
3    // inputPIN is
4    // secret
5    if (x < 1234)
6      print(0);
7    y = x;
8    print(y);
9    // public output
```

(2)

```
1  void main():
2    fork thread_1();
3    fork thread_2();
4  void thread_1():
5    x = input();
6    print(x);
7  void thread_2():
8    y = inputPIN();
9    x = y;
```

(3)

```
1  void main():
2    fork thread_1();
3    fork thread_2();
4  void thread_1():
5    doSomething();
6    print('G');
7  void thread_2():
8    y = inputPIN();
9    while (y != 0)
10     y--;
11   print('I');
```

Abb. 2: Kleine, aber typische Beispiele für Lecks. (1) explizite und implizite Lecks. (2) possibilistisches Leck. (3) probabilistisches Leck.

(4)

```
1  void main():
2    h = inputPIN();
3    l = 2;
4    // l is public
5    x = f(h);
6    y = f(l);
7    print(y);
8
9  int f(int x)
10   {return x+42;}
```

(5)

```
1  void main():
2    fork thread_1();
3    fork thread_2();
4  void thread_1():
5    l = 42;
6    h = inputPIN();
7  void thread_2():
8    print(l);
9    l = h;
```

(6)

```
1  void main():
2    L = 0;
3    read(H2);
4    while (H2>0) H2--;
5    fork thread_1();
6    fork thread_2();
7  void thread_1():
8    l = 42;
9    h = inputPIN();
10 void thread_2():
11   print(l);
12   l = h;
```

Abb. 3: Sichere Programme, die durch mangelnde Analyse-Präzision Fehlalarme auslösen. (4) kontext-insensitive Analyse verschmilzt die beiden `f`-Aufrufe. (5) fluss-insensitive Analyse ignoriert die Anweisungsreihenfolge in `thread_2`. (6) klassische "low-deterministic security" verbietet jeden Low-Nichtdeterminismus.

garantien für schwierige Beispiele aus der Literatur herleiten, sowie ein prototypisches E-Voting System – inclusive Kryptographie – als sicher nachweisen [KTG12]. Die vollständige Analyse des bekannten HSQLDB Datenbanksystems dauerte einige Stunden auf einem Standard-PC.

3 Probabilistische Nichtinterferenz

IFC für sequentielle Programme muss explizite und implizite (Vertraulichkeits-)Lecks erkennen, die entstehen, wenn (Teile von) geheimen Daten an öffentliche Variablen kopiert werden; bzw. wenn geheime Daten den Kontrollfluss beinflussen. (siehe Beispiel (1) in Abbildung 2). IFC für nebenläufige Programme mit gemeinsamem Speicher ist eine viel größere Herausforderung, da zusätzliche possibilistische oder probabilistische Lecks entdeckt werden müssen, die durch Interleaving des Schedulers entstehen können. In Beispiel (2) führt die Scheduling-Reihenfolge 5,8,9,6 zum Ausgeben der geheimen PIN. In Beispiel (3) wird die PIN zwar nie ausgegeben, jedoch erhöht eine große PIN die Wahrscheinlichkeit, dass `'GI'` und nicht `'IG'` ausgegeben wird – denn die Laufzeit der Schleife hängt

von der PIN ab. Ein solches probabilistisches Leck kann einem geduldigen Angreifer viel Informationen über geheime Werte liefern; Abschnitt 5 zeigt eine größere Fallstudie.

IFC Algorithmen prüfen *Nichtinterferenz*. Nichtinterferenz bedeutet grob gesagt, dass öffentlich sichtbares Programmverhalten nicht durch geheime Werte beeinflusst werden darf. Alle korrekten Nichtinterferenz-Kriterien sind hinreichend, aber nicht notwendig, so dass zwar garantiert alle Lecks gefunden werden, es aber zu Fehlalarmen kommen kann. Die *Präzision* von IFC-Algorithmen ist deshalb wichtig, um Fehlalarme zu reduzieren (gänzlich eliminieren kann man sie aus Entscheidbarkeitsgründen nicht [Satz von Rice]). Gute IFC nutzt moderne Programmanalyse, um Präzision und Skalierbarkeit zu maximieren. So ist es etwa wichtig, die Reihenfolge von Anweisungen zu beachten ("Fluss-Sensitivität"), ebenso wie verschiedene Aufrufe derselben Funktion zu unterscheiden ("Kontext-Sensitivität") und verschiedene Objekte derselben Klasse zu unterscheiden ("Objekt-Sensitivität"). Beispiele (4) und (5) in Abbildung 3 zeigen Programme, die bei fluss- bzw kontext-insensitiver Analyse zu Fehlalarmen führen. Es ist aber absolut nichttrivial, skalierende Analysen zu entwickeln, die diese (und andere) Sensitivitäten haben; die lange Zeit populären Sicherheits-Typsysteme jedenfalls sind zumeist weder fluss- noch kontextsensitiv.

Probabilistische Nichtinterferenz verlangt nicht nur, dass es keine expliziten oder impliziten Lecks gibt; sondern zusätzlich, dass die *Wahrscheinlichkeit* für jedes öffentliche Verhalten durch geheime Werte nicht beeinflusst wird (vgl. Beispiele (3) und (6)). Gerade die bekannten Kriterien und Algorithmen für probabilistische Nichtinterferenz unterscheiden sich auf subtile Weise, mit jeweiligen Vor- und Nachteilen. So ist das einfachste Kriterium, die "Low-security observable determinism" (LSOD) [RWW94, ZM03] relativ einfach zu prüfen und funktioniert für jeden Scheduler, verbietet aber jeden öffentlichen Nichtdeterminismus – auch wenn dieser nie geheime Information preisgeben kann. Kriterien, die diese Einschränkung nicht haben (u.a. [SS00, Smi06, MS10]), stellen dafür – teils unrealistische – Anforderungen an den Scheduler oder das Programm. Obwohl LSOD seinerzeit nicht populär war, entschieden wir uns, JOANA auf LSOD-Basis zu realisieren – in der später bestätigten Hoffnung, dass sich durch Einsatz moderner Programmanalyse noch wesentliche Verbesserungen erzielen lassen.

JOANA verwendet deshalb sog. Programmabhängigkeitsgraphen (PDGs), die fluss- und kontextsensitiv sind, jedoch deutlich komplexer als Typsysteme. Heutige PDGs, gekoppelt mit Points-to Analyse und Exception Analyse sind sehr präzise, behandeln volles Java, und skalieren bis mehrere hunderttausend LOC. PDGs und ihr Zusammenhang mit IFC wurden oft beschrieben, so dass wir hier darauf verzichten – für Details siehe [HS09, GS15]. Untersuchungen zur Präzision und Skalierbarkeit der PDG-basierten IFC finden sich in [GS15, Ham10].

4 Das iRLSOD Kriterium

Bereits 2003 gab Zdancewicz ein einfach zu handhabendes Kriterium für LSOD an [ZM03]:

1. es darf keine expliziten oder impliziten Lecks geben;
2. keine öffentliche (low) Operation wird durch ein Data Race beeinflusst;

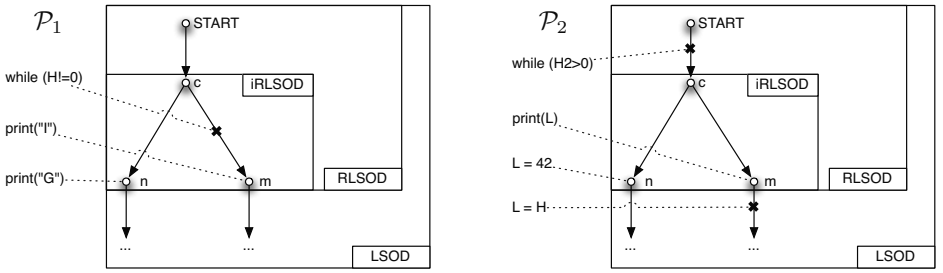


Abb. 4: Visualisierung von LSOD vs. RLSOD vs. iRLSOD. Skizziert sind die Kontrollflussgraphen für Beispiel (3) und Beispiel (6). n und m produzieren Low-Nichtdeterminismus, c ist der gemeinsame Dominator. LSOD verbietet jeglichen Low-Nichtdeterminismus; RLSOD erlaubt Low-Nichtdeterminismus der nicht von High-Anweisungen erreichbar ist; iRLSOD erlaubt Low-Nichtdeterminismus der von High-Anweisungen erreichbar ist, sofern letztere vor dem gemeinsamen Dominator liegen. Die markierten Bereiche zeigen, wo Low-Nichtdeterminimsmus weiterhin verboten ist; man erkennt dass iRLSOD wesentlich mehr Low-Nichtdeterminismus ohne Fehlalarme erlaubt als RLSOD oder gar LSOD.

3. es gibt keinen Low-Nichtdeterminismus.

Der letzte Punkt verbietet, dass jegliche Low-Operationen parallel ausgeführt werden – sogar wenn das nachweisbar sicher wäre. Giffhorn entdeckte in seiner Dissertation [Gif12], dass Fluss-Sensitivität der Schlüssel zu einer präzisen Implementierung des Zdancewicz-Kriteriums ist, und bewies die Korrektheit seines Verfahrens. Insbesondere können alle drei Bedingungen in natürlicher Weise mit PDGs und nebenläufigen Kontrollflussgraphen überprüft werden [GS15]; zusätzlich wird eine präzise Points-to Analyse und eine präzise "May happen in parallel" (MHP) Analyse verwendet. Damit profitiert IFC von den langjährigen, internationalen Anstrengungen zur Konstruktion präziser PDGs für volles Java.

Der Vollständigkeit halber sei Giffhorns LSOD Kriterium und Korrektheitstheorem angegeben, ohne dass wir auf die komplexen Begrifflichkeiten, den Beweis, oder die algorithmischen Herausforderungen eingehen können (siehe [GS15]). Das Kriterium realisiert die 3 Zdancewicz-Bedingungen mittels des PDG und Slicing. $cl(n)$ ist die Klassifikation (Low oder High) des PDG-Knotens bzw der Anweisung n, $BS(n)$ ist der (fluss-, kontext-, objekt-sensitive) Rückwärts-Slice im PDG für n.

Theorem. LSOD und also probabilistische Nichtinterferenz gilt, wenn für alle PDG Knoten n, n', n'' gilt:

1. $cl(n) = L \wedge cl(n') = H \implies n' \notin BS(n)$
2. $MHP(n,n') \wedge \exists v \in def(n) \cap (def(n') \cup use(n')) \wedge cl(n'') = L$
   $\implies n, n' \notin BS(n'')$
3. $MHP(n,n') \implies cl(n) = H \vee cl(n') = H$

Giffhorn entdeckte auch, dass man die Bedingung 3. abschwächen und gewissen Low-Determinismus erlauben kann. Wenn nämlich dieser Low-Nichtdeterminismus nicht im

Kontrollflussgraph von High-Operationen aus erreichbar ist, kann er kein probabilistisches Leck hervorrufen ("RLSOD"-Kriterium: Relaxed LSOD). So ist etwa der Low-Nichtdeterminismus in Beispiel (5) unschädlich, weil zwar die Reihenfolge von Anweisungen 5. und 8. i.a. nichtdeterministisch (nämlich schedulerabhängig) ist, diese Anweisungen aber nicht von High-Anweisungen (insbesondere 6.) erreichbar sind. Andererseits ist in Beispiel (3) der Low-Nichtdeterminismus von Anweisungen 6. und 11. sehr wohl von Anweisungen 8./9. erreichbar, das Programm deshalb unsicher (siehe Abschnitt 3). Beispiel (6) ist hingegen sicher, denn zwar ist der Low-Nichtdeterminismus in Anweisungen 8./11. von den High-Anweisungen 3./4. erreichbar, jedoch beinflussen 3./4. die Anweisungen 8. und 11. in identischer Weise, so dass ein Angreifer trotz des 8./11. Nichtdeterminismus nichts über `H2` erfahren kann!

Das letzte Beispiel führt auf das neue iRLSOD Kriterium. Es reicht zu prüfen, ob der High-Einfluss auf den Low-Nichtdeterminismus vor oder hinter dem *gemeinsamen Dominator* [Aho86] zweier nichtdeterministischer Low-Anweisungen n, m stattfindet. Wie in Abbildung 4 dargestellt, ist etwa im Beispiel (3) eine High-Anweisung auf einem der beiden Kontrollfluss-Zweige vom gemeinsamen Dominator $c = cdom(n,m)$ zu n bzw. m, also *hinter* c. Hingegen gilt für Beispiel (6), dass die High-Anweisung *vor* c liegt. iRLSOD verbietet nur solche High-Anweisungen, die hinter c, aber vor n bzw. m liegen. Da der gemeinsame Dominator in der Praxis weit hinter dem START-Knoten liegt, ergibt sich eine stark reduzierte Zahl von Fehlalarmen durch Low-Nichtdeterminismus; gleichzeitig werden weiterhin alle probabilistischen Lecks garantiert erkannt.

Der Vollständigkeit halber geben wir die modifizierte Bedingung (3) nebst Korrektheitsaussage für iRLSOD an; für Bedingung (2) kann man eine analoge iRLSOD-Version angeben. Details und Beweis finden sich in [BGH+16].

Theorem. Probabilistische Nichtinterferenz gilt, wenn für alle PDG Knoten n, n', n'' LSOD-Bedingung (1) und (2) gilt, und ferner:

3. $MHP(n,n') \wedge cl(n) = cl(n') = L \wedge c = cdom(n,n')$
$\implies \forall n'' \in \big((c \rightarrow^*_{CFG} n) \cup (c \rightarrow^*_{CFG} n')\big) : cl(n'') = L$

5 Fallstudie: E-Voting

In diesem Abschnitt zeigen wir die iRLSOD-Analyse eines experimentellen E-Voting Systems, das in Zusammenarbeit mit R. Küsters et al. entwickelt wurde. Die E-Voting Software soll beweisbar eine fundamentale kryptographische Eigenschaft aufweisen, nämlich die sog. *Computational Indistinguishability*. Um dies durch IFC zu unterstützen, entwickelten Küsters et al. einen Ansatz, bei dem zunächst kryptographische Funktionen durch eine Leerimplementierung ersetzt werden, die lediglich dieselben Schnittstellen wie die echte Krypto-Software aufweist. Kann man dann mit IFC Nichtinterferenz beweisen (d.h. es gibt keine Lecks zwischen Klartext, Schlüssel und Schlüsseltext), so gilt nach einem Theorem von Küsters die "computational indistinguishability" für die echte Implementierung [KTG12, KSTG14]. Für eine rein sequentielle Implementierung konnte JOANA problemlos Nichtinterferenz nachweisen.

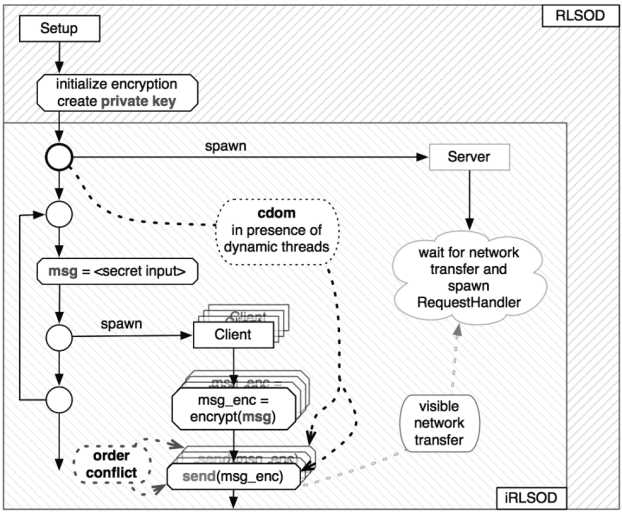


Abb. 5: Struktur des Kontrollflussgraphen für Server-Client basierten Nachrichtenaustausch mit Multithreading.

Eine neuere Implementierung verwendet eine Client-Server Architektur, und besteht aus einigen hundert LOC in 16 Klassen. Der interprozedurale Kontrollfluss ist in Abbildung 5 dargestellt; Abbildung 6 zeigt die relevanten Abschnitte des Quelltextes. Der Hauptthread startet in Klasse `Setup` in Zeile 3ff: Zunächst wird die Verschlüsselung durch Erzeugen von geheimem und öffentlichem Schlüssel initialisiert; dann wird ein einzelner `Server` Thread gestartet, bevor die Hauptschleife beginnt. In der Hauptschleife wird jeweils eine geheime (High) Nachricht gelesen, dann wird jeweils ein `Client` Thread gestartet, der diese verschlüsselt und über das Internet zum Server sendet. In diesem Beispiel gibt es also viele Instanzen des `Client` Threads, je einen pro Nachricht. Die Computational Indistinguishability verlangt, dass der Angreifer nicht zwischen verschiedenen (verschlüsselten) Nachrichteninhalten unterscheiden kann, und nach Küsters Theorem reicht es, Nichtinterferenz zu prüfen.

In diesem Beispiel sind High: (1) der Wert des `secret_bit` (Zeile 3), welches das Wahlverhalten codiert und über den Inhalt der Nachricht entscheidet,[5] und (2) der private Schlüssel (Zeile 33). Beide wurden in JOANA mittels einer `@Source` Annotation als HIGH markiert. Gemäß Angriffsmodell kann der Angreifer die versendeten Nachrichten lesen (aber nicht entschlüsseln), deshalb wurden Aufrufe von `sendMessage` (Zeile 73f) als LOW `@Sink` markiert. JOANA schließt explizite oder implizite Lecks sofort aus, findet im RLSOD-Modus aber zwei probabilistische Lecks – von denen sich eines mittels iRLSOD als Fehlalarm entpuppt.

Das erste potentielle Leck entsteht dadurch, dass die geheime Schlüsselinitialisierung evtl. den `sendMessage` Aufruf beeinflusst. Zur iRLSOD-Analyse dieses Lecks sei angemerkt,

[5] In diesem Beispiel wird angenommen, dass der Wähler nur zwischen 2 Parteien wählen kann, so dass ein Bit für das E-Voting reicht.

```
1  public class Setup {
2
3    public static void setup(@Source boolean secret_bit) { // HIGH input
4      // Public-key encryption functionality for Server
5      Decryptor serverDec = new Decryptor();
6      Encryptor serverEnc = serverDec.getEncryptor();
7      // Creating the server
8      Server server = new Server(serverDec, PORT);
9      new Thread(server).start();
10
11     // The adversary decides how many clients we create
12     while (Environment.untrustedInput() != 0) {
13       // determine the value the client encrypts:
14       // the adversary gives two values
15       byte[] msg1 = Environment.untrustedInputMessage();
16       byte[] msg2 = Environment.untrustedInputMessage();
17       if (msg1.length != msg2.length) { break; }
18
19       byte[] msg = new byte[msg1.length];
20       for(int i = 0; i < msg1.length; ++i)
21         msg[i] = (secret_bit ? msg1[i] : msg2[i]);
22
23       // spawn new client thread
24       Client client = new Client(serverEnc, msg, HOST, PORT);
25       new Thread(client).start();
26     }
27   }
28 }
29
30 public class KeyPair {
31   public byte[] publicKey;
32   @Source
33   public byte[] privateKey; // HIGH value
34 }
35
36 public final class Decryptor {
37
38   private byte[] privKey;
39   private byte[] publKey;
40   private MessagePairList log = new MessagePairList();
41
42   public Decryptor() {
43     // initialize public and secret (HIGH) keys
44     KeyPair keypair = CryptoLib.pke_generateKeyPair();
45     publKey = copyOf(keypair.publicKey);
46     privKey = copyOf(keypair.privateKey);
47   }
48
49   ...
50
51 }
52
53 public class Client implements Runnable {
54
55   private byte[] msg;        private Encryptor enc;
56   private String hostname;   private int port;
57   ...
58
59   @Override
60   public void run() {
61     // encrypt
62     byte[] mgs_enc = enc.encrypt(msg);
63
64     // send
65     long socketID = socketID = Network.openConnection(hostname, port);
66     Network.sendMessage(socketID, mgs_enc);
67     Network.closeConnection(socketID);
68   }
69 }
70
71 public class Network {
72
73   @Sink // LOW output
74   public static void sendMessage(long socketID, byte[] msg) throws NetworkError {
75     ...
76   }
77   ...
78 }
```

Abb. 6: Relevante Sourcecode-Ausschnitte des E-Voting Systems

dass durch die dynamischen Threads die Dominatorberechnung angepasst werden muss: der Dominator aller `sendMessage` Aufrufe (die als potentiell nichtdeterministische Low-Operationen zu betrachten sind) ist der Kopf der While-Schleife. Vor dieser Schleife liegt die Schlüsselinitialisierung, die als High klassifiziert ist (denn sie wird durch den geheimen Schlüssel beeinflusst).[6] Nach iRLSOD liegt sie aber vor dem Dominator und kann kein probabilistisches Leck verursachen – das "alte" RLSOD würde hier einen Fehlalarm erzeugen. Die Schlüsselinitialisierung ist also sicher, und iRLSOD – aber nicht RLSOD – kann das garantieren.

Ein weiteres probabilistisches Leck entsteht aber dadurch, dass der `encrypt` Aufruf vom geheimen Wert `msg` beeinflusst wird. Zwar kann durch die Krypto-Leerimplementierung kein Fehlalarm eines expliziten Lecks erfolgen, und auch eine Deklassifikation ist nicht notwendig. Jedoch kann nicht ausgeschlossen werden, dass die *Laufzeit* des Aufrufs von `msg` abhängt, so dass ein Leck analog Beispiel (3) entstehen könnte. Denn wenn die `encrypt` Laufzeit vom Nachrichteninhalt abhängt, wird das Scheduling statistisch[7] eine bestimmte Reihenfolge der Nachrichten hervorrufen, durch die ein Angreifer evtl. Rückschlüsse auf den Inhalt ziehen kann. JOANA entdeckt dieses probabilistische Leck, weil die iRLSOD-Bedingung verletzt ist: Der `encrypt`-Aufruf ist als High klassifiziert (vgl. Fußnote), liegt aber zwischen Dominator (Schleifenkopf) und Low-nichtdeterministischem `sendMessage`-Aufruf.

JOANA brauchte 5 Sekunden zur Analyse des gesamten Quelltextes. JOANA zeigt im Quelltext die nichtdeterministischen Anweisungen an nebst Hinweis, dass der Nichtdeterminismus sichtbares Verhalten beeinflussen kann. Der Prüfingenieur könnte daraufhin in diesem Beispiel allerdings entscheiden, dass das Leck nicht gefährlich ist! Wenn er z.B. weiss, dass der Nichtdeterminismus nur durch die `encrypt` Laufzeiten entstehen kann, und wenn er garantieren kann, dass die `encrypt` Laufzeit unabhängig vom Nachrichteninhalt ist, darf er das Leck ignorieren. Solche manuellen Entscheidungen (die auch als Deklassifikation im Quelltext vermerkt werden können) erfordern allerdings genaue Programmkenntnis und größte Vorsicht.

6 Verwandte Projekte

JIF [Mye99] ist eins der ältesten IFC Werkzeuge, aber verwendet einen speziellen Java-Dialekt und braucht viele Annotationen. Das kommerziell erfolgreichste IFC Tool ist vermutlich TAJ / Andromeda [TPF+09], das in der IBM AppScan Suite verwendet wird. Das Tool erkennt nur explizite Lecks, skaliert aber in den MLoc Bereich. FlowDroid [ARF+14] kann keine probabilistischen Lecks behandeln, bietet aber guten Support für den Android Life Cycle und Reflection. Für JavaScript wurden Kombinationen von statischer und dynamischer IFC vorgeschlagen [BRGH14].

[6] JOANA klassifiziert Anweisungen, die von High-Eingaben direkt oder indirekt kontroll- oder datenabhängig sind, automatisch ebenfalls als High [HS09].

[7] (i)RLSOD setzt einen echt probabilistischen Scheduler voraus [BGH+16], so dass sich tatsächlich bei verschiedener Laufzeit der `encrypt` Aufrufe statistische Aufruf-Muster bilden, die der Angreifer bobachten kann – auch wenn er den Nachrichteninhalt nicht entschlüsseln kann.

7 Schluss

In diesem Artikel haben wir neue Ergebnisse des JOANA-Projektes skizziert, und das neue iRLSOD-Kriterium zur probabilistischen Nichtinterferenz an Beispielen erklärt. Die tatsächliche Formalisierung nebst Korrektheitsbeweis ist sehr viel komplexer – hier seien nur die Probleme der Terminations-Sensitivität oder der Scheduler-Abhängigkeit erwähnt. Details zu PDG-basierter IFC und (i)RLSOD finden sich in [HS09, GS15, BGH+16].

Heute ist JOANA eines der wenigen international verfügbaren IFC Werkzeuge, das volles Java incl. Threads mit hoher Präzision behandeln kann und an realen Fallstudien erprobt wurde. Wir glauben, dass wir die weitverbreitete Skepsis gegenüber dem "Low-Deterministic Security" Ansatz mit iRLSOD widerlegt haben. Der nächste Schritt wird sein, JOANA an industrieller Software zu erproben.

Danksagung. JOANA wird seit vielen Jahren von der DFG unterstützt (insbesondere durch DFG SPP 1496 "Reliably secure software systems"), sowie vom BMBF im Rahmen des Software-Sicherheits-Kompetenzzentrums KASTEL.

Literaturverzeichnis

[Aho86] Ullman Aho, Sethi. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, 1986.

[ARF+14] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Traon, D. Octeau und P. McDaniel. FlowDroid: precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android apps. In *Proc. PLDI*, Seite 29, 2014.

[BGH+16] Joachim Breitner, Jürgen Graf, Martin Hecker, Martin Mohr und Gregor Snelting. On Improvements Of Low-Deterministic Security. In *Proc. Principles of Security and Trust POST*, 2016. to appear.

[BRGH14] A. Bichhawat, V. Rajani, D. Garg und C. Hammer. Information Flow Control in WebKit's JavaScript Bytecode. In *Proc. Principles of Security and Trust POST*, Seiten 159–178, 2014.

[GHMS15] Jürgen Graf, Martin Hecker, Martin Mohr und Gregor Snelting. Checking Applications using Security APIs with JOANA. Juli 2015. 8th International Workshop on Analysis of Security APIs.

[Gif12] Dennis Giffhorn. *Slicing of Concurrent Programs and its Application to Information Flow Control*. Dissertation, Karlsruher Institut für Technologie, Fakultät für Informatik, Mai 2012.

[Gra16] Jürgen Graf. *Information Flow Control with System Dependence Graphs — Improving Modularity, Scalability and Precision for Object Oriented Languages*. Dissertation, Karlsruher Institut für Technologie, Fakultät für Informatik, 2016. Forthcoming.

[GS15] Dennis Giffhorn und Gregor Snelting. A New Algorithm For Low-Deterministic Security. *International Journal of Information Security*, 14(3):263–287, April 2015.

[Ham10] Christian Hammer. Experiences with PDG-based IFC. In F. Massacci, D. Wallach und N. Zannone, Hrsg., *Proc. ESSoS'10*, Jgg. 5965 of *LNCS*, Seiten 44–60. Springer-Verlag, Februar 2010.

[HS09] Christian Hammer und Gregor Snelting. Flow-Sensitive, Context-Sensitive, and Object-sensitive Information Flow Control Based on Program Dependence Graphs. *International Journal of Information Security*, 8(6):399–422, December 2009.

[KSTG14] Ralf Küsters, Enrico Scapin, Tomasz Truderung und Jürgen Graf. Extending and Applying a Framework for the Cryptographic Verification of Java Programs. In *Proc. POST 2014*, LNCS 8424, Seiten 220–239. Springer, 2014.

[KTG12] Ralf Küsters, Tomasz Truderung und Jürgen Graf. A Framework for the Cryptographic Verification of Java-like Programs. In *Computer Security Foundations Symposium (CSF), 2012 IEEE 25th*. IEEE Computer Society, Juni 2012.

[MGH15] Martin Mohr, Jürgen Graf und Martin Hecker. JoDroid: Adding Android Support to a Static Information Flow Control Tool. In *Proceedings of the 8th Working Conference on Programming Languages (ATPS'15)*, Lecture Notes in Informatics (LNI) 215. Springer Berlin / Heidelberg, Februar 2015. to appear.

[MS10] Heiko Mantel und Henning Sudbrock. Flexible Scheduler-Independent Security. In *Proc. ESORICS*, Jgg. 6345 of *LNCS*, Seiten 116–133, 2010.

[Mye99] Andrew C. Myers. JFlow: practical mostly-static information flow control. In *POPL '99: Proceedings of the 26th ACM SIGPLAN-SIGACT symposium on Principles of programming languages*, Seiten 228–241, New York, NY, USA, 1999. ACM Press.

[RWW94] A. W. Roscoe, Jim Woodcock und L. Wulf. Non-Interference Through Determinism. In *ESORICS*, Jgg. 875 of *LNCS*, Seiten 33–53, 1994.

[Smi06] Geoffrey Smith. Improved typings for probabilistic noninterference in a multi-threaded language. *Journal of Computer Security*, 14(6):591–623, 2006.

[SS00] Andrei Sabelfeld und David Sands. Probabilistic Noninterference for Multi-Threaded Programs. In *Proc. CSFW '00*, Seite 200, Washington, DC, USA, 2000. IEEE Computer Society.

[TPF+09] Omer Tripp, Marco Pistoia, Stephen J. Fink, Manu Sridharan und Omri Weisman. TAJ: effective taint analysis of web applications. In *Proc. PLDI*, Seiten 87–97, 2009.

[WLS09] Daniel Wasserrab, Denis Lohner und Gregor Snelting. On PDG-Based Noninterference and its Modular Proof. In *Proc. PLAS '09*. ACM, June 2009.

[ZM03] Steve Zdancewic und Andrew C. Myers. Observational Determinism for Concurrent Program Security. In *Proc. CSFW*, Seiten 29–43. IEEE, 2003.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 23

Distributed Evolutionary Fuzzing with Evofuzz

Fabian Beterke[1]

Abstract: This paper describes the design of a tool (called *Evofuzz*) that implements the technique of evolutionary (or coverage-guided) fuzzing in a scalable, distributed manner. The architecture, design-choices and implementation specifics of this tool are examined, explained and criticized. After outlining possible improvements and future work that is not yet completed, the paper finishes by presenting the results from fuzzing real-world programs and explains how to recreate them using the provided tool.

Keywords: Fuzzing, Fuzzer, Evolutionary, Distributed Computing, Evofuzz, Security Research, Memory Corruption

1 Motivation

After the concept of fuzzing [Mi88] was introduced by Barton Miller (University of Wisconsin) et al. in 1988, the techniques and its variations have led to the discovery of innumerable issues, particularly in the field of memory-corruption vulnerabilities. Since then, the technique has evolved drastically into numerous branches like the original random fuzzing, generative fuzzing and mutation-based fuzzing. While those techniques have been implemented, analyzed and evaluated numerous times before, the idea of guiding the fuzzing process (i. e. the search through the target-program's state-space for unexpected/undefined behavior) by code-coverage is relatively new, the first public mention being with the goal of exhaustively testing the program without a brute-force approach of simply generating every possible input. While evolutionary fuzzing has been proposed in 2007 already by DeMott et al. [DEP07], no public and general-purpose implementation has been available until in February 2014, M. Zalewski published American Fuzzy Lop [Za15] (subsequently referred to as AFL). AFL has since showed a tremendous story of success, uncovering hundreds of bugs in otherwise well researched and widespread software which is due to its careful balance between a capable concept of mutating data in an evolutionary fashion and maintaining a sense of pragmatism, e. g. in terms of always keeping performance (measured in runs of the target program per second) the major focus because more runs always mean a higher likelihood of discovering crashes. A restricting factor for this metric however is the single-threaded design of AFL: Concurrently running it on a few dozen machines can be a very cumbersome task since it can only be achieved by copying over temporary files from one node to another and running each process independently, and since the target programs are run on a single core only, no performance gains can be observed from the use of multicore systems. A customizable, distributed-by-design, evolutionary general-purpose fuzzer would solve those problems which motivated the author to start working on a scalable, coverage-guided fuzzing architecture.

[1] Universität Hamburg, SVS, Vogt-Kölln-Straße 30, 22527 Hamburg, 2beterke@informatik.uni-hamburg.de

2 Background: Evolutionary Fuzzing

While of course the ultimate measurement for a fuzzer is the number of unique crashes it found, there are a few other metrics to orientate at when designing a new fuzzer, the most important one being code-coverage. Since the likelihood of finding issues increases with the amount of actual code that is being tested, covering as much functionality as possible is a valid (and measurable) metric for the success of a fuzzer. One of the most prevalent questions when designing a new fuzzer is whether the testcases for the target program should be generated entirely from scratch or as mutations of already existing inputs – the so-called *corpus* [MP07] – to achieve a high coverage and find numerous bugs. For a general purpose fuzzer that needs no further description of the input data than the samples in the corpus, many different mutation strategies (like e. g. flipping bits, inserting or removing bytes and performing arithmetic operations on chunks of the input) are needed in order to be flexible enough to handle various formats well. If however no big and coverage-rich corpus is available, the options can become sparse very quickly and hand-writing a new grammar for each format to be (generatively) fuzzed is a very time-consuming task.

Another option emerges if the problem of fuzzing a program for memory-corruption bugs is seen from another angle: essentially, fuzzing is nothing else than a search through the state-space of the program for crashing states. The formerly mentioned metric of code-coverage can be seen as a distance function: The more functionality (coverage) a testcase triggers, the higher the likelihood of discovering vulnerabilities when mutating it. The term *evolutionary fuzzing* (coined by DeMott et al. with their talk at Defcon 2007 [DEP07]) refers to this technique which can be summarized by the following algorithm:

1. Seed input queue with corpus
2. Pop a single testcase from the queue, generate mutations and execute
3. If one of the mutated inputs led to execution of previously unexecuted code: add testcase to input queue
4. Repeat from 2. until input queue is empty

This algorithm has the property of exploring the target program's state-space through evolving the input given to the program. While of course, starting from a good corpus drastically improves the overall performance of the fuzzer (i. e. in terms of achieved input complexity vs. CPU cycles spent), an experiment [Za14] by M. Zalewski (the author of AFL) showed that even an absolutely meaningless input with basically no coverage can be used as a corpus for a good evolutionary fuzzer to produce coverage-rich inputs. In this extreme example, Zalewski used only one file containing the ASCII-string "hello" as an initial corpus for fuzzing the target, a JPEG decoder. After just one day of fuzzing, AFL generated a wide array of actually valid JPEG files that make use of a variety of JPEG features – an excerpt of this experiment's results is shown in Figure 1.

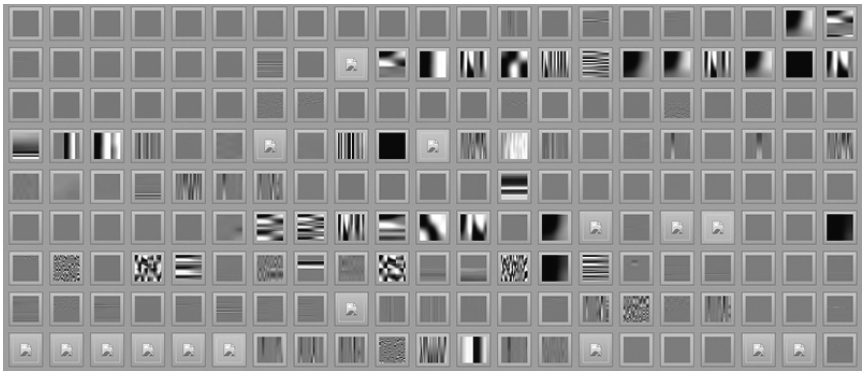


Fig. 1: JPEG images generated mutationally by AFL [Za14]

3 Measuring Code Coverage with LLVM's SanitizerCoverage

In order to develop such a coverage-guided fuzzer, it is crucial to gather data about the program's actual execution flow under a certain input, giving insight about the fuzzer's code-coverage. The LLVM Compiler-Suite offers a solution to this problem with a compile-time instrumentation called *Sanitizer Coverage* [Te15], or Sancov for short. At compile time, the tool's user has to choose between two relevant modes of code-coverage as they are offered by Sancov:

Basic-Block Level:
A widespread way to track coverage is to instrument every generated conditional jump instruction, effectively splitting the code into basic blocks of execution. Those blocks are the unit for the block level coverage method which can be illustrated by the following, C-like pseudocode:

```
1 int reset(int *x) {
2     if(x)            // A
3         *x = 0;      // B
4     return 1;        // C
5 }
```

While line 1 and 5 are irrelevant since they contain no control-flow altering logic, basic blocks in the assembly code generated from that snippet would be line 2 (A), 3 (B) and 4 (C). The comparison in block A takes place once the function is called, the write-operation in block B is dependent on the former comparison and the return instruction in block C is again executed unconditionally. Sancov offers output for this measurement in two different formats which may be chosen at runtime through an environment variable, either a list of instruction-pointers for basic-blocks (i. e. their first instruction's address) or a ASCII-bitmap where each index corresponds to a basic block and a 0 or 1 character indicates whether the block was executed or not. This level of instrumentation has a runtime overhead of around 30% on average, depending on the properties of the instrumented program.

Edge Level:
When imagining the execution flow as a graph of basic-blocks, a shortcoming of simple block-based coverage measurement emerges. While the coverage actually measured by block-level instrumentation relies on executed basic blocks (the graph's vertices), the edges in this graph represent state transitions which are more meaningful than pure basic blocks because a bug can seldom be found in a basic block alone but rather emerges from a faulty state transition. In the example code, without prior knowledge about the logic utilized by the program it is not clear whether a state transition from A to C happened only via B or directly, without invoking B first. Those edges are called critical edges, and by introducing dummy basic-blocks into those edges, the execution graph's edges can be instrumented analogous to basic-block coverage. This again increases the runtime overhead to 40% on average.

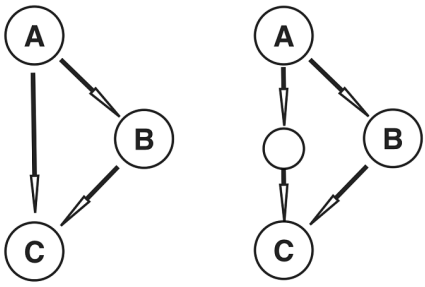


Fig. 2: Comparison between Block Coverage (left) and Edge Coverage (right)

4 Distributed Fuzzing: Problems and Challenges

The high-level organisation of the fuzzer follows a master-worker pattern in which a master-node is responsible for multiple workers, but multiple masters can be present if needed. While the master-node keeps the state of the overall fuzzing process, governs the testcase-queue and analyzes e. g. logfiles of crashes or code-coverage information, the worker-node's purpose is to execute testcases specified by the master and report the execution results back.

In order to avoid complex synchronization scenarios between a multitude of worker-nodes, one of the fuzzer's major design goals is to keep as little state as possible and do it on the master-node(s) only, wherever possible in the (itself distributed) persistent storage engine. This design allows the fuzzer to be scaled infinitely in theory. Since worker- and master-nodes are connected via a network-connection, one of the aims for a high throughput of target-program executions is to keep the network's bandwidth-utilization as low as possible. The problem that must be solved to achieve this is to actually produce the mutations on the worker-node itself and execute them right away while still centrally guiding the fuzzing process by e. g. specifying the testcase to be mutated, which mutation should be used as well as the amount of mutated testcases to produce.

5 Designing Evofuzz: Architecture and Implementation

Master

The master server is written in Python 2.7 and is responsible for central, stateful tasks. Its main functionalities are bundled into modules, each run in their own thread and communicate via a simple, message-passing based IPC implemented using the *threading.Queue* class. The persistence layer is built around Redis [SN10] (short for "Remote Dictionary Server"), an in-memory database with optional persistence to disk that offers simple but effective datastructures like e. g. hashes, lists and sets. Redis itself is highly scalable, on the one hand in a redundant way with many synchronized nodes propagating updates through the cluster to form a high-availability system with failover capabilities (called Redis Sentinel) and in a sharded fashion on the other hand (Redis Cluster), theoretically allowing the fuzzer to utilize up to $2^{14} = 16384$ nodes as one big, distributed in-memory database (this number is an artifact of Redis Cluster's implementation, however the authors recommend 1000 nodes as a reasonable upper limit for stable operation).

Notable modules of the master include the *ProvisionServer* which provides (target program) binaries, testcases and configuration to the workers on request and the *ReportServer* which generates a job description for the next chunk of work to be sent out to the worker while at the same time receiving their reports. The latter are redirected to the *ReportAnalyzer* module which is responsible for implementing the evolutionary fuzzing-logic of enqueuing mutations of testcases that lead to previously unexecuted code. Additionally, signatures of the newly found basic-blocks (or edges if that coverage method is used) are pushed to the *UpdatePublisher*, a caching layer that propagates those newly found signatures to the worker-nodes in order to prevent a high network utilization by useless re-submissions.

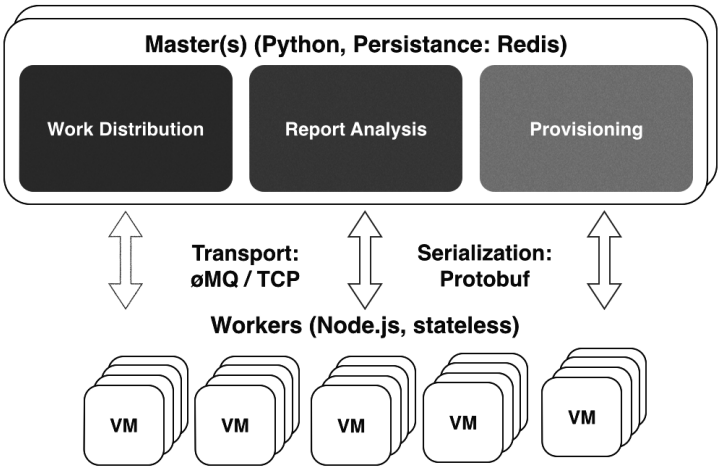


Fig. 3: Master Architecture and Overview

Worker

Since the main purpose of a worker node is to mutate data and actually run the target-program as fast as possible and at the same time, keep as little state as possible, the author chose to develop it in Javascript (JS) using NodeJS. The main reason for this choice is the comparably easy implementation of asynchronous, heavily callback-oriented actions: nearly all APIs to the underlying Operating System's features like spawning processes or doing network-/fileystem-IO are available as fully asynchronous versions by default since JS embraces this style of programming. Also, NodeJS is powered by Google's V8 JS interpreter which makes use of just-in-time compilation, leading to negligible performance disadvantages compared to native languages like C or C++ while being suited for explorative, rapid development due to JS being a memory-safe and loosely typed language.

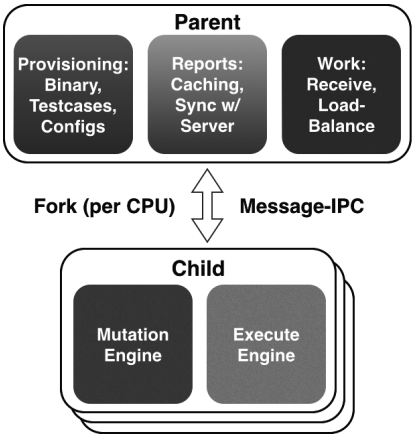


Fig. 4: Worker Architecture

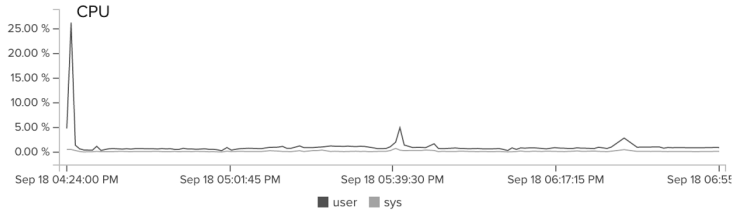
As shown in the figure above, the worker architecture is again separated into a parent and one or more multiple child processes. The parent process manages the connection to the master-node and initializes the fuzzing-loop by requesting a provision from the master, effectively setting up the (virtual) machine this process is running on by e. g. transferring the target program and it's command line arguments and environment variables from the master. Other responsibilities include buffering reports (coverage and crash information) before sending them to the server in chunks, subscribing to the *UpdatePublisher* and maintaining a cache to prevent redundant reports as well as load-balancing incoming work between children.

Those children-processes implement the actual generation of testcases, generating all possible mutations of a certain kind for a single testcase before starting to use the next mutation. Furthermore, those processes execute the target program with the mutated input, collect the Sancov-output and probe for the existence of a crashlog after each run and eventually send the results of those operations up to the parent process for reporting to the master.

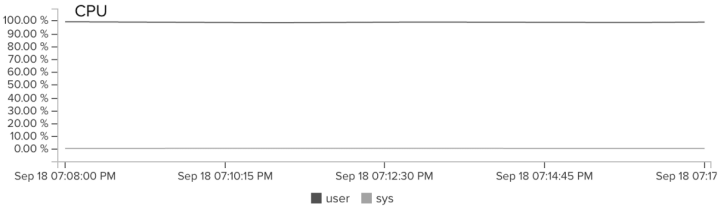
6 Performance and Scalability

The design outlined in the previous section allows the fuzzer to be scaled arbitrarily in theory while this has practical limitations of course (e. g. the resources of the machines used, the network connection between them or database capacities). The actual performance in terms of execution runs per second however is highly dependent on the inner loop, i. e. the execution of a single testcase. Because of the use of AddressSanitizer (another sanitizer from the LLVM suite that instruments memory accesses to detect e. g. out-of-bounds reads) and Sancov in conjunction with the lack of in-process fuzzing mechanisms for the sake of genericness, this performance-critical part of the fuzzing loop is not (yet) as optimized as it could be – currently, the fuzzer exhibits between 120 and 150 runs per second per core on average with a Intel Xeon CPU (E5-2630L) clocked at 2.4GHz (AFL for comparison reaches around 800 executions per second on this CPU but can only utilize one core), obviously varying wildly for different target programs. As stated previously, this shortcoming can be overcome easily by scaling horizontally, i. e. adding more worker nodes to the cluster due to the fuzzer's distributed design. The reason why this is a necessary requirement for a long-running fuzzer is that performance may not only be measured in runs per second or the percentage of basic-blocks discovered but also in terms of cost-effectiveness: It is a lot cheaper to rent 100 separate machines with 1 core each for an hour than it is to rent a single 100-core machine while at the same time, the latter is also more prone to error due to being a single point of failure. Furthermore, effectiveness is influenced by resource utilization: It is mandatory for the fuzzer to utilize all available ressources at all times, meaning 100% CPU utilization at every point in time for a worker node and preferably very little load on a master node so that new work may be handed out without latency, thus preventing idle times (as shown in Figure 5).

Fig. 5: CPU utilization on different node types



(a) Master (1 CPU, 512M, 30 Workers)



(b) Worker

7 Usage

Prior to configuring and starting the fuzzer, the user needs to compile the desired target with AddressSanitizer and Sancov enabled. This boils down to the use of 2 simple command line flags in a recent version of *clang*:
`clang target.c -fsanitize=address -fsanitize-coverage=X`
where `X` is replaced by a number to indicate which coverage instrumentation should be used, i. e. 2 for block-level and 3 for edge-level coverage. The actual usage of the fuzzer can be summarized into 3 high-level steps:

1. Initialize Redis with config and corpus using project-specific `initRedis.sh` (located in `misc/`)
2. Run `master.py` on master node
3. Start desired number of worker nodes and run `worker.js <master IP>`

This process can of course be automated for specific cloud-computing hosters, making use of their APIs for e. g. fully automated deployment of an arbitrarily sized fuzzing cluster. While the results outlined in this paper have mainly been produced by using *Digital Ocean*, a deployment on e. g. *Amazon Web Services* would work analogous. After a successful fuzzing run, crashes may be extracted from the database by simply running the utility in `misc/analyzeCrash.py`.

```
sh-3.2$ tar -xzf fuzz.tgz && cd fuzz && ls
README.md       master          misc            protos          worker
sh-3.2$ cd misc && sh initRedis.sh
OK
OK
(integer) 1
(integer) 1
(integer) 1
(integer) 1
sh-3.2$ cd .. && virtualenv master; cd master && . bin/activate; make &> /dev/null
New python executable in master/bin/python
Installing setuptools, pip...done.
(master)sh-3.2$ python master.py
[ProvisionServer] got PROVISION request
[ProvisionServer] got STATE request
[ControlServer] worker connected, total: 1
[ReportServer] initiating
[Bitflip] <DEBUG> initiated
[ProvisionServer] got TESTCASE request
[Bitflip] <DEBUG> next -> True
[Bitflip2] <DEBUG> initiated
[ReportAnalyzer] 1 new blocks/edges found!
[Bitflip2] <DEBUG> next -> True
[ReportAnalyzer] \o/ got a new crash \o/
[ReportAnalyzer] 1 new blocks/edges found!
[Bitflip4] <DEBUG> initiated
[Bitflip4] <DEBUG> next -> True
[Byteflip] <DEBUG> initiated
[Byteflip2] <DEBUG> initiated
[ControlServer] worker 26 disconnected, requeing 4 job(s)
^CTerminating, sending kill-signal...
Terminated: 15
(master)sh-3.2$
```

```
sh-4.3$ tar -xzf fuzz.tgz && cd fuzz
sh-4.3$ cd worker && sh prepareFS.sh && make > /dev/null
npm WARN package.json evoFuzz@0.0.1 No repository field.
npm WARN package.json evoFuzz@0.0.1 No README data
sh-4.3$ node worker.js 10.0.2.2
coverage initiated, pathlen: 9
Worker online
Worker online
Worker online
Worker online
bitflip
bitflip
worker #2 got work, 1000 testcases generated
worker #1 got work, 1000 testcases generated
bitflip
worker #3 got work, 48 testcases generated
bitflip2
worker #4 got work, 1000 testcases generated
bitflip2
worker #3 got work, 1000 testcases generated
got update! added 0 hashes
=== 183.06666666666666 runs/sec ===
received crash from worker 2!
got update! added 0 hashes
bitflip2
worker #2 got work, 48 testcases generated
bitflip4
worker #4 got work, 1000 testcases generated
bitflip4
worker #2 got work, 1000 testcases generated
bitflip4
worker #3 got work, 48 testcases generated
byteflip
worker #1 got work, 256 testcases generated
byteflip2
worker #3 got work, 256 testcases generated
^C
sh-4.3$
```

Fig. 6: Set up of the fuzzer + CLI interface while running

8 Results

The three main objectives of the evolutionary fuzzer as outlined in this paper are a design that enables it to be scaled horizontally on commodity hardware, the ability to discover more complex inputs leading to previously unseen behaviour as well as finding crashes, obviously. The design choices that allow the fuzzer to achieve the first of the mentioned goals have been explained in depth in the previous parts, the fulfilment of the second objective can be measured easily by keeping track of the number of discovered basic blocks in an exemplary target program (`libharfbuzz` in this case, an advanced font parsing library used e. g. in Google Chrome). Also, the general effects of lowering the time needed to trigger the execution of those new basic blocks by simply adding more workers to the cluster has been successfully demonstrated (see Figure 7, comparability is given since only deterministic mutations have been used for this measurement). In a few test runs conducted with various numbers of worker nodes, the fuzzer was able to discover crashes in numerous targets, including `libharfbuzz`, `libfreetype`, `speexdec` and `p7zip` just to name the most important ones. It has been shown that the tool fulfills all of it's objectives in a sufficient, though still improvable way.

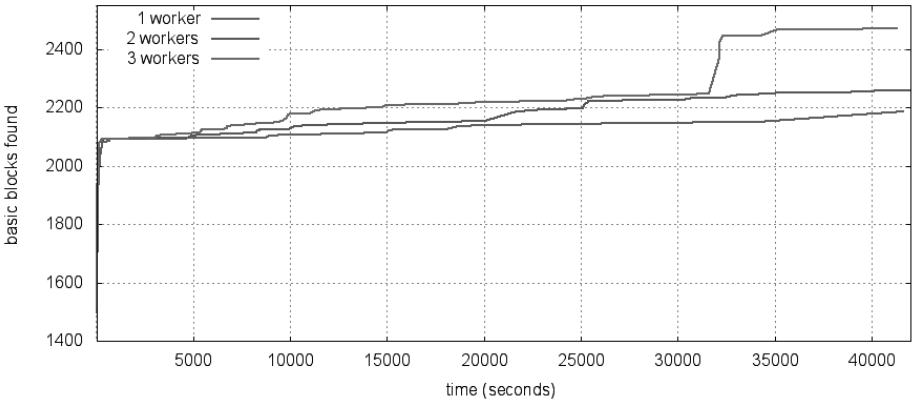


Fig. 7: Discovery of basic blocks with different numbers of workers in `libharfbuzz`

9 Criticism and future work

A few known issues in the actual implementation (rather than the design) exist at the current point of development that have not yet been tackled because of time constraints. The main problem the author currently faces in terms of performance improvements is the inner-loop optimization in order to make the fuzzer yield more executions per second per core: the possibilities of the implementation in Node.js have been exhausted to a maximum, meaning that further improvements like generic in-process fuzzing or forkless execution as well as saving copy- and process-spawning overhead would need a rewrite of the worker-implementation in C/C++ or another hardware-close language that compiles to native code and allows more fine granular access to the underlying operating system's API.

Another major improvement would be automatic testcase minimisation, the so called pruning: once a testcase leading to new behaviour is found, it could be repeatedly trimmed in size at different positions while examining which is the smallest mutation (by number of bytes) that still leads to the exact same behaviour of the target program.

An improvement that was planned but not yet implemented is the automated reproducibility check for crashes: A crash that can not immediately be reproduced is worthless for the fuzzing process since there may be numerous indeterministic reasons why a program could crash without the crash being related to the actual input which is why those inputs should be instantly discarded rather than having the user figure out which crashes can be actually reproduced. Finally, the software could be made more usable by a more intuitive interface, e. g. a Web-Interface.

10 Conclusions

The author proposed a design for a distributed, evolutionary fuzzing solution that may be used with arbitrary target programs (as long as they are available in source code and compile in clang / clang++) and as such, is generic. An upper limit for its scalability is yet to be found and the fuzzer behaved as expected even in tests with more than 50 worker nodes, yielding over 100,000 executions per second in total. The tool introduced in this paper may be used by developers or security researchers alike to fuzz for security bugs, even if no sophisticated corpus is available for the target program. Also, the codebase is kept quite small as well as comparably clean and modular, enabling interested parties to improve or customize the solution for their individual needs and expectations.

References

[DEP07] DeMott, Jared; Enbody, Richard; Punch, William: , Revolutionizing the Field of Grey-box Attack Surface Testing with Evolutionary Fuzzing. `https://www.blackhat.com/presentations/bh-usa-07/DeMott_Enbody_and_Punch/Presentation/bh-usa-07-demott_enbody_and_punch.pdf`, 2007. Talk at Blackhat.

[Mi88] Barton Miller: Random fuzz testing. `http://pages.cs.wisc.edu/~bart/fuzz/`, accessed 8.12.2015.

[MP07] Miller, Charlie; Peterson, Zachary N. J.: , Analysis of Mutation and Generation-Based Fuzzing, 2007.

[SN10] Salvatore Sanfilippo: Redis. `http://redis.io`, accessed 8.12.2015.

[Te15] The Clang Development Team: Sanitizer Coverage. `http://clang.llvm.org/docs/SanitizerCoverage.html`, accessed 8.12.2015.

[Za14] Michal Zalewski: Pulling JPEGs out of thin air. `http://lcamtuf.blogspot.de/2014/11/pulling-jpegs-out-of-thin-air.html`, accessed 8.12.2015.

[Za15] Michal Zalewski: American Fuzzy Lop's technical details. `http://lcamtuf.coredump.cx/afl/technical_details.txt`, accessed 8.12.2015.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 33

Attacks on Fitness Trackers Revisited: A Case-Study of Unfit Firmware Security

Jakob Rieck[1]

Abstract: Fitness trackers – wearables that continuously record a wearer's step count and related activity data – are quickly gaining in popularity. Apart from being useful for individuals seeking a more healthy lifestyle, their data is also being used in court and by insurance companies to adjust premiums. For these use cases, it is essential to ensure authenticity and integrity of data. Here we demonstrate a flaw in the way firmware for Withings' *Activité* is verified, allowing an adversary to compromise the tracker itself. This type of attack has so far not been applied to fitness trackers. Vendors have started mitigating previous attacks, which manipulated data by interfering with wireless channels, or by physically moving the tracker to fool sensors. Hardware similarities amongst different trackers suggest findings can be transferred to other tracker as well.

Keywords: Fitness Tracker, IoT, Wearable Security, Withings, Firmware Modification, Reverse Engineering

1 Introduction

Fitness trackers – wearable devices that capture various statistics about their wearer and try to motivate physical activity – have become increasingly popular in recent times. According to Canalys, a market research firm, wearable fitness bands grew 684% worldwide in the first half of 2014 compared to the first half of 2013 [Ca14]. Estimations for future growth vary wildly, indicating the sector is not yet well understood and possibly grows much faster than older, more traditional sectors [We14].

Fitness trackers are typically wrist-worn devices that collect and send data such as the wearer's step count via a user's device to the companies' server. Existing research indicates that generated data, despite being noisy, can be used to answer intimate questions, such as if a person recently quit smoking [KTK14] or whether two humans work together [TKS13].

Insurance companies have started to offer benefits for active customers who choose to share their fitness data with them [Be15]. Data generated by fitness trackers is used in court rooms and can decide the outcome of a trial [Hi15, Ol14]. Both use-cases raise questions about how the data's *integrity* and *authenticity* can be ensured. If a vendor fails to address these issues, insurance fraud starts to become a problem and fabricated evidence ends up deciding cases in court. Existing technical attacks try to manipulate data in transit and can be effectively mitigated by affected vendors. Firmware attacks, seemingly overlooked in this context, pose a new challenge, as they violate trust placed in hardware.

[1] University of Hamburg, FB Informatik, Security in Distributed Systems Group, Vogt-Kölln-Straße 30, 22527 Hamburg, 2rieck@informatik.uni-hamburg.de

This paper contributes to the field as follows: In the first part, a general architectural model of fitness trackers is presented and existing attacks are outlined. The analysis shows that the security of firmware updates is crucial, but has not received much attention in the literature. Therefore, the second part presents a novel firmware modification attack against one specific fitness tracker.

The first part consists of Sections 1 to 5. In Section 2, related work is reviewed. Section 3 will go into more detail about how a fitness tracker works, which technologies are typically used and how the various devices work together, resulting in a general model of fitness trackers. Section 4 characterises the adversary used for the remainder of this paper. Previously published attacks on fitness trackers are surveyed in Section 5. The main contribution of this work – a proof-of-concept firmware attack against Withings' collection of *Activité* trackers – is presented in Section 6, followed by a conclusion in Section 7.

2 Related Work

Related work can be divided into two important categories: Firstly, works in the narrow context of fitness trackers focussing on attacking data integrity, as well as discussions about the privacy and cultural issues surrounding their ever expanding presence are surveyed. After that, selected works regarding firmware security and firmware modification attacks are summarised.

Kawamoto and Tsubouchi demonstrate that data from accelerometers, as also used in fitness trackers, can be used to answer highly personal questions about a user [KTK14, TKS13], identifying the need for adequate privacy protections for users. Paul et al. discuss the privacy policies of various vendors, identifying with whom companies intend to share the data produced and what rights a user retains over their own data. They identify a number of worrying statements regarding the ownership and commercial usage of user data [PI14]. In 2013, Rahman et al. examined Fitbit's communication protocol and found several weaknesses, e.g. unencrypted and unauthenticated data upload, allowing for easy data manipulation. They devised *FitLock*, a "defense system" for the vulnerabilities discussed [RCB13]. Zhou et al. followed up on this work by identifying shortcomings in *FitLock*, but did not propose their own set of modifications to fix the aforementioned issues [ZP14]. Reexamining their work, in 2014, Rahman et al. published a paper detailing weaknesses in both Fitbit's and Garmin's communication protocol, enabling them to send fake data to the server and inject fake data into the fitness tracker. They devised *SensCrypt*, a "secure protocol for managing low power fitness tracker" [RCT14]. Symantec published a white paper in 2014, highlighting issues regarding the handling of private information by fitness trackers. They were able to track individuals using cheap, off-the-shelf hardware by simply connecting to fitness trackers on the street.

Costin et al. in 2014 performed a large scale analysis of the security of embedded firmware. They presented a correlation technique that allows for propagation of existing vulnerability information to similar firmware that has previously not been identified as vulnerable. One downside of their approach is that it is rather granular and cannot identify previously

unknown firmware modification attacks [Co14b]. Cui et al. present a general discussion of firmware modification attacks and then focus their work on HP's LaserJet printers, identifying an issue allowing an adversary with print permissions to update the affected printer using modified firmware [CCS13]. Coppola in 2013 studied the firmware update procedure of Withings' *WS-30* wireless scale. He combined hardware and software reverse engineering to exploit a critical flaw in the update process, allowing him to upload arbitrary firmware to the scale [Co13].

Coppola's unpublished work on Withings' *WS-30* is closest to our work in terms of content. Because fitness trackers process much more sensitive information, use a different communication architecture compared to Withings' scales, and it has been more than 2 years since Coppola demonstrated his work, firmware security on Withings' platform should have improved, warranting an analysis of the security of firmware updates in the context of fitness trackers. Such an analysis in this context has so far been overlooked in academia and, to the best of our knowledge, has not been published before.

3 Architecture Types

Fitness trackers are small, light-weight pieces of technology typically worn on the wrist. There are however also trackers that can be worn on the shoe, the waist, or the upper arm.

Priced anywhere from $15 (Xiaomi MiBand) to over $150 (Fitbit Charge HR), all trackers have the same core function: Counting steps. In fact, as can be seen from Tab. 1, the only sensor found in every fitness tracker listed is an accelerometer. This sensor is used to infer a number of data points during the day such as number of steps taken, calories burned, distance travelled, as well as time slept during the night. High-end trackers such as Fitbit's *Charge HR* typically feature a small display that shows time, daily statistics and notifications from a connected phone. Additionally, this device also includes an altimeter to measure the number of flights of stairs and an optical heart rate sensor to continuously measure the wearer's heart rate [Fi15]. Many trackers also feature a vibration motor to silently wake up the wearer.

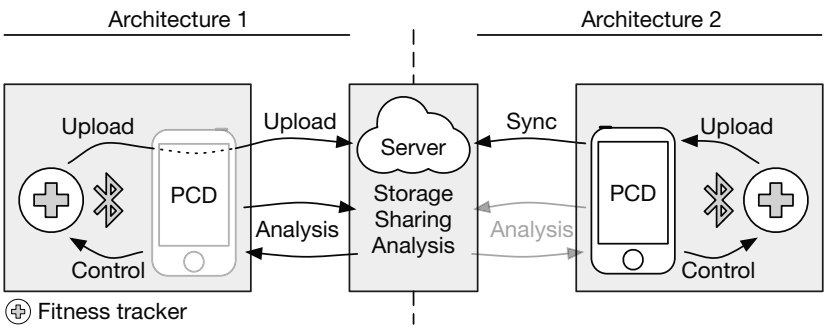
Tab. 1: Various fitness trackers along with the sensors built in.

	Display	Accelerometer	Heartrate	Altimeter	Vibration motor
Fitbit Charge HR	✓	✓	✓	✓	✓
Misfit Flash	✗	✓	✗	✗	✗
Xiaomi MiBand	✗	✓	✗	✗	✓
Withings Activité	✗	✓	✗	✗	✓

Due to power and space constraints, all vendors have chosen to use Bluetooth Low Energy (BLE) instead of WiFi as their communication channel. After pairing the tracker with a suitable smartphone, tablet or computer (hereafter referred to as "Personal Communcations Device" or simply PCD), data is transferred between tracker and an app on the PCD via Bluetooth Low Energy. For more details regarding Bluetooth Low Energy, refer to [De14] and follow-up works where the general architecture and communication patterns are summarised.

Broadly speaking, as shown in Fig. 1, there are two different viable communication architectures.

Fig. 1: Communication architectures of fitness trackers



The left-hand side depicts an architecture where the software running on the PCD is not capable of interpreting the data received and acts as a "dumb" pipe between tracker and backend server. The PCD in turn has to query the server for "insights" and visualisations of the data. Thus, in this architecture, the server serves a very important role. Most notably, *Architecture 1* enables tracker and server to communicate end-to-end encrypted. In contrast, the right-hand side visualises an architecture where the software on the PCD is capable of interpreting data sent by the tracker, enabling visualising the data without calling on the server. In such an architecture, the server is still used for syncing, sharing and more advanced analysis.

Both architecture types – or slight variations of them – are used in practice: Fitbit, following a string of attacks abusing unencrypted data communications between PCD and server, now seems to be using an approach much closer *Architecture 1*, where the applications itself only passes on the data [Ap15, Cy14]. In contrast, based on the fact that Withings' *Health Mate* application can show detailed statistics even without a working Internet connection, this platform is much more similar to the second architecture.

4 Adversary Characterisation

This section presents a concise, informal description of the adversaries' motives and capabilities. Specifying a formal *attacker model* makes little sense in this context, as fitness trackers themselves are not rigorously formalised.

From the point of view of the tracker's vendor – which is considered to be a benign actor – the tracker's *user* is the primary adversary. Exercising control over both tracker and PCD and being motivated by possible financial gain, this adversary has both means and motives to try to attack the system.

The user is considered to have full control over their PCD – including the ability to execute arbitrary code (and therefore being able to tamper with the vendor's code). This control

allows for passive sniffing attacks on both wireless channels – Bluetooth and WiFi – to eavesdrop on communications between server, PCD, and tracker. Active attacks, such as manipulations of traffic by injecting or modifying data or denying service altogether are possible as well. Indeed, even TLS encrypted communications can be intercepted and manipulated by using a web debugging proxy (see also Sect. 5).

The adversary is thought to perform only software-based attacks that do leave visible traces of tampering on the target tracker itself – Hardware based attacks, such as physically dumping firmware or data, are often very challenging, as microcontrollers use internal memory and ship with code protection features. Hardware reverse engineering is therefore explicitly *not* considered. By default, the adversary is only able to use the tracker as intended, and tries to abuse his control over the other parts of the system to manipulate data or to gain full control of the tracker itself.

Other adversary models were discussed in the past: A malicious vendor would trivially be able to create malicious firmware or could modify data at will. The analysis of third-party adversaries manipulating communication channels (e.g. WiFi and / or Bluetooth) has been part of previous work (see Sect. 2) and would require a thorough discussion of the communication medium itself, which is beyond the scope of this work.

5 Attacks on Data Integrity

This section surveys existing attacks on data integrity. So far, attacks have mainly attacked exposed air channels: Both the connection between PCD and server and the local connection between PCD and tracker can be targeted. This section investigates these issues in more detail.

The first point of attack, Internet traffic, was used by practically all previous work on this matter: In their 2013 paper [RCB13], Rahman et al. investigate the state of security for the Fitbit *One*. They found that all traffic between basestation and the vendor's website was unencrypted, allowing them to upload modified data to the Fitbit website. Following up on their work, they identified a similar vulnerability in Garmin's web service [RCT14]. Using TLS should fix this problem, however, for the adversary as outlined in Section 4, even a correctly implemented TLS application does not present a significant hurdle. Generally, the workflow to inspect and modify TLS traffic on one's own devices goes as follows: A web debugging proxy with TLS support, e.g. *Fiddler* [Te03] or *mitmproxy* [Co14a], is installed. A custom root certificate is installed on the user's PCD allowing the proxy to generate TLS certificates on the fly for each requested website. Because the device in question trusts the custom root certificate, it also trusts all generated TLS certificates. This is a side-effect of the way the trust model works for TLS and is *not* strictly speaking a vulnerability. If employed, *certificate pinning* can be bypassed by patching the binaries, as demonstrated in practice [Kr15, Ri12]. For the considered adversary, TLS therefore does not pose a significant challenge.

For the Bluetooth link, similar attacks are possible. Because the considered adversary is one of the communicating parties, she does not need to break any existing protocols. In

some instances, even an adversary that does not control the user's PCD can attack or at least monitor the Bluetooth traffic of fitness trackers: As detailed in [RCT14], missing authentication allows one to harvest information of bystanders or to write bogus data to the tracker itself. Reports indicate that such attacks are still possible today [BWL14].

The last attack vector is of physical nature, where the tracker is manipulated in a way that recorded activities are not actually performed by the legitimate owner. Such "mule-attacks" [RCB13] make the tracker record a correct measurement of fabricated facts, violating the security property *veracity* as proposed by Gollmann [Go12]. This type of attack is not specific to fitness trackers: In much the same way, temperature readings can be manipulated and smoke alarms set off by temporarily manipulating the sensor's local environment. *Unfit-Bits* (http://www.unfitbits.com/) is one parody website solely devoted to detailing ways of fooling fitness trackers [MB15].

6 Security of Firmware Updates: A case-study of Withings' *Activité*

So far attacks discussed in this paper have attacked two aspects: The first kind of attack interferes with WiFi and Bluetooth channels to manipulate readings after they have been generated on the tracker. Secondly, "mule-attacks" manipulate the local environment to fool sensors to record steps and activities that are not performed by the legitimate owner. Both attacks allow a determined adversary to achieve the goal of data manipulation. However, both techniques require the adversary to actively manipulate the tracker over the desired period of time. Vendors have already started addressing problems abused by attacks discussed previously. Fitbit for example seems to have completely re-designed their communication infrastructure to prevent attacks abusing exposed air channels [Ap15]. We expect that vendors will also try to detect and prevent mule attacks in the future.

In contrast, attacks on firmware – software running directly on the microprocessor embedded in these devices – are much more capable. Such attacks can also be used to manipulate the step count and activity levels, simply by outputting incorrect information in the first place. Unlike other attacks, a firmware attack requires larger up-front costs, but scales much better: After the initial injection, no more unusual interactions with the tracker are required. In this sense, they are more convenient for an adversary. Furthermore, every single aspect of the device that is accessible by firmware can also be controlled by the adversary, including Bluetooth functionality and data storage. Firmware attacks are interesting to other adversaries apart from the user, as they offer features such as stealthiness and persistence that none of the other attacks offer.

In the following, a proof-of-concept firmware attack against Withings' *Activité* fitness tracker is presented.

6.1 Methodology

The most straight-forward method to inject custom firmware is to abuse official update procedures. In order to exploit a flaw in this process, two basic requirements have to be

met. First off, the firmware has to be transferred to the device over a (wireless) channel under the control of the adversary. This holds true, because the user controls the PCD interacting with the tracker via Bluetooth. Secondly, there have to be flaws in the verification of firmware updates. Because the PCD is under control of the adversary, effective verification and authentication can only be performed on the tracker itself.

To develop the proof-of-concept attack presented here, a number of challenges had to be overcome: The (unencrypted) firmware had to be dumped and analysed. A new image had to be crafted and re-signed to pass verification on the tracker. Finally, the modified firmware update had to be injected, preferably using the official update procedure.

To distribute updates, Withings uses the *Health Mate* application on iOS and Android. All analysis presented in this paper was conducted using *Health Mate* version 2.6.2 on Apple's iOS. When an update for the tracker is available, the user is notified on the main screen of the application. The app automatically downloads new firmware in the background, so that when the user initiates the update, the firmware file is transmitted wirelessly over Bluetooth onto the tracker, where it is subsequently installed. The downloaded file is not encrypted, which was immediately obvious because debug strings are readily available. To view and modify traffic between PCD and server, *mitmproxy*, a free web debugging proxy with TLS support, was used [Co14a].

The firmware analysis consisted of disassembling the firmware in *IDA Pro* [He]. To use *IDA* effectively, some information such as the processor architecture used and the load address of the input file in memory are required.

Information regarding the processor can usually be found by examining the user's manual, publicly available FCC documents or public teardowns. Indeed, one public teardown [In15] identifies a variant of the *nRF51822* as being used. This is a low-power SoC (System on a Chip) combining Bluetooth Low Energy with a 32-bit ARM Cortex-M0 which is capable of executing code that uses the *ARMv6-M* instruction set [Se15, AR09].

The second step, identifying a firmware's load address in memory, is much more challenging. Fortunately, previous work by Coppola addresses this problem [Co13]. A list of candidate load addresses can be built up by iteratively trying different base addresses and, judging by the number of strings referenced in the disassembly using this base address, choosing the best match. Because two distinct binaries were packed into one file and both files were loaded at different addresses in memory, the aforementioned approach only yielded the correct load address for the first, larger image. Therefore, as a third step, the second image's load address was determined by identifying almost identical debug functions in both binaries that referenced identical strings. An approach to finding base addresses automatically makes use of jump tables and indirect jumps and is described in detail in [Sh15].

Manual reverse engineering is a tedious, error-prone task. With no symbol information available, as is the case for the task at hand, the analyst is left with a difficult job. Fortunately, large parts of Withings' firmware contain debug strings, making it much easier to identify code sections of interest.

6.2 Results

By reverse engineering the code within firmware images, it was possible to identify a header structure at the beginning of each firmware update that is used for firmware verification. Figure 2 shows parts of the reconstructed header structures. After the header's version number – which so far has always been `1` for all firmware updates examined – and it's length – which does not include the final `table_checksum` field – are two structures describing each of the two images that are bundled within an update, including their respective offset, length and version number. There's also a checksum field that is validated by computing a checksum over the actual contents of the image, not the header itself. Finally, the so-called `table_checksum` is computed over the contents of the header. Based on debug strings in the update, the two binaries are internally referred to as "app" and "bootloader". Together, these binaries make up the firmware of the *Activité*.

Fig. 2: Header structure definition

```
struct activite_header_t {
    uint16_t table_ver;
    uint16_t table_len;
    image_t  images[2];
    uint32_t table_checksum;
};
```

```
struct image_t {
    uint16_t identifier;
    // ...
    uint32_t checksum;
    uint32_t version;
};
```

Checksums – *CRC-32* is used here – can safeguard against transmission errors, but they cannot be used to verify the *authenticity* of an update. An adversary can just replace the checksums after modifying the original firmware to circumvent all checks on the tracker itself. There is nothing inside the update file that could be used to verify the authenticity of an update. This reveals a major design flaw in Withings' firmware security, which relies solely on the paradigm of *Security through Obscurity*.

Crafting a modified firmware image is not much of a challenge. Any modification can be used to demonstrate the issue. Modifications such as manipulating the step count are comparatively simple, whereas complex, malicious modifications are harder and require substantial (re-)engineering resources. At this point, all modifications have to be performed by patching the firmware file directly. In the future, it might be possible to adopt work by Wang et al. to introduce more complex modifications by recompiling the disassembly, rather than patching individual bytes [WWW15].

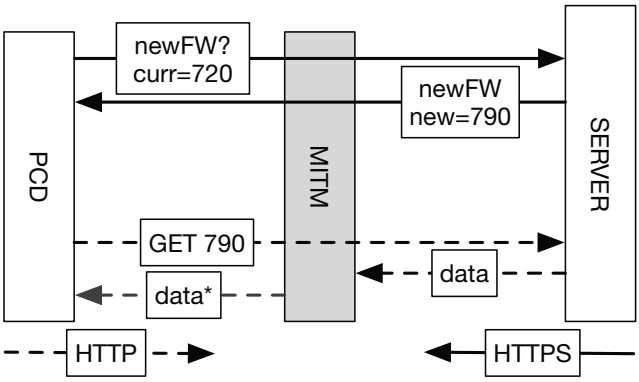
As a proof-of-concept, the version number inside the update was modified, the firmware "resigned" and injected (see next paragraph for details). The tracker successfully accepted the update and transmitted the modified version number to the *Health Mate* application.

6.3 Impact

The described attacked can be performed by any adversary capable of manipulating HTTP traffic – provided an official update is available.

On each start of the *Health Mate* application, a *GET* request is made to Withings' server to check for the availability of new firmware. The response includes a URL to new firmware, if one is available. Figure 3 depicts the update process along with the actions of a third-party capable of manipulating plain HTTP traffic. What can be seen is that such an adversary is still able to serve malicious updates, provided an official update is available. The considered adversary (see Sect. 4) is more powerful: She can additionally fake the availability of new firmware using a TLS proxy and thus does not need to wait for official updates to inject custom firmware.

Fig. 3: Attacking update procedure



The discovered flaw and resulting attack apply to both Withings' *Activité* and *Activité Pop* and presumably *Activité Steel* (At the time of writing, no device for testing was available). Analysis of the firmware of Withings' *Pulse Ox* reveals the same underlying design defect. Coppola showed in 2013 that Withings' *WS-30* wireless scale also did not verify the authenticity of firmware updates [Co13]. Because ARM's Cortex-M0 cores lack any kind of privilege separation [AR09, p.33], malicious updates have control over the whole chip and can block further legitimate updates or dynamically patch new firmware updates for persistence.

The results are potentially also transferrable to trackers of other vendors because of the similarities in the design of fitness trackers – Misfit's *Flash* for example uses the same microprocessor as Withings' *Activité*.

6.4 Countermeasures

The main problem identified in the previous section is the lack of authenticity checks in Withings' firmware updates.

There are two routes to improve the current situation: As a first step, it would be wise for Withings to remove all debug strings from production firmware. While this does not fix the underlying issue, it makes it a lot harder to reverse engineer firmware. Furthermore, encrypting the firmware and decrypting the firmware on the tracker itself – *not* on the PCD

which is controlled by an adversary – would mean an adversary has to physically attack the tracker to obtain the decrypted firmware.

Fundamentally though, the authenticity of firmware has to be verified by the tracker itself. Two options come to mind: Either a unique, symmetric key is shipped with every device which then can be used to verify the firmware's *MAC* (message authentication code), or a public key is embedded on the device or in the firmware itself, which is then used to authenticate subsequent updates. The symmetric key required to verify a *MAC* has to be unique, otherwise a single compromised device jeopardises the security of all devices. Because each device would then have to receive individual firmware updates, shipping a public key inside the firmware might be the more cost-effective alternative. Public key cryptography however poses a significant challenge for slow, battery-constrained devices. These challenges are beyond the scope of this paper.

Finally, it is important to mention that properly authenticated firmware updates, as important as they are, do not make firmware "secure". Vulnerabilities in the code itself can still be exploited to attack firmware.

Countermeasures proposed here are solely meant to improve the security of firmware updates. Mitigations implemented on the PCD (e.g. certificate pinning) can only help defend against adversaries who do not control the PCD and are therefore not mentioned here.

7 Conclusion

Being able to trust the authenticity and integrity of fitness data generated by trackers is essential for a number of use cases such as presenting the data as evidence in court and adjusting premiums for insurants. A survey on existing attacks on data integrity for trackers reveals that thus far, firmware level attacks were overlooked by the academic community.

In this paper, a vulnerability in the firmware update process for Withings' *Activité* was identified that can be used to inject malicious code into the tracker. This is all the more worrying considering that due to limitations imposed by hardware design, there is no privilege separation on this particular tracker family.

As of now, modifications to Withings' firmware are fairly limited in scope and impact, due to the fact that manual binary patching has to be used. In the future, work by Wang et al. might allow for much easier development of custom firmware, by recompiling disassembled firmware. Reverse engineering and documenting more functions in official updates – not just parts that are used for verifying updates – could allow for much more extensive modifications.

Most importantly, further research investigating other fitness trackers with respect to firmware security is needed.

References

[Ap15] Apvrille, A.: Fitness Tracker: Hack In Progress. https://www.hackinparis.com/talk-2015-fitness-tracker (Accessed on November 26, 2015), 2015.

[AR09] ARM Limited.: Cortex-M0 – Technical Reference Manual, revision r0p0. http://infocenter.arm.com/help/topic/com.arm.doc.ddi0432c/DDI0432C_cortex_m0_r0p0_trm.pdf (Accessed on November 22, 2015), 2009.

[Be15] Bernard, T. S.: Giving Out Private Data for Discount in Insurance. http://www.nytimes.com/2015/04/08/your-money/giving-out-private-data-for-discount-in-insurance.html (Accessed on November 21, 2015), 2015.

[BWL14] Barcena, M.; Wueest, C.; Lau, H.: How safe is your quantified self. https://www.blackhat.com/docs/eu-14/materials/eu-14-Wueest-Quantified-Self-A-Path-To-Self-Enlightenment-Or-Just-A-Security-Nightmare-wp.pdf (Accessed on November 22, 2015), 2014.

[Ca14] Canalys: Wearable band shipments rocket by 684%. http://www.canalys.com/newsroom/wearable-band-shipments-rocket-684 (Accessed on December 13, 2015), 2014.

[CCS13] Cui, A.; Costello, M.; Stolfo, S.: When Firmware Modifications Attack: A Case Study of Embedded Exploitation. In: NDSS. The Internet Society, 2013.

[Co13] Coppola, M.: Summercon 2013: Hacking the Withings WS-30. http://poppopret.org/2013/06/10/summercon-2013-hacking-the-withings-ws-30/ (Accessed on November 26, 2015), 2013.

[Co14a] Cortesi, A.: mitmproxy. https://mitmproxy.org/ (Accessed on November 22, 2015), 2014.

[Co14b] Costin, A.; Zaddach, J.; Francillon, A.; Balzarotti, D.: A Large-Scale Analysis of the Security of Embedded Firmwares. In: 23rd USENIX Security Symposium (USENIX Security 14). USENIX Association, San Diego, CA, pp. 95–110, August 2014.

[Cy14] Cyr, B.; Horn, W.; Miao, D.; Specter, M.: Security Analysis of Wearable Fitness Devices (Fitbit). https://courses.csail.mit.edu/6.857/2014/files/17-cyrbritt-webbhorn-specter-dmiao-hacking-fitbit.pdf (Accessed on December 11, 2015), 2014.

[De14] Decuir, J.: Introducing Bluetooth Smart: Part 1: A look at both classic and new technologies. Consumer Electronics Magazine, IEEE, 3(1):12–18, Jan 2014.

[Fi15] Fitbit: Fitbit Announces Global Availability of Highly Anticipated Fitbit Charge HR and Fitbit Surge. https://investor.fitbit.com/press/press-releases/press-release-details/2015/Fitbit-Announces-Global-Availability-of-Highly-Anticipated-Fitbit-Charge-HR-and-Fitbit-Surge/default.aspx (Accessed on November 21, 2015), 2015.

[Go12] Gollmann, D.: Veracity, plausibility, and reputation. In: Information Security Theory and Practice. Security, Privacy and Trust in Computing Systems and Ambient Intelligent Ecosystems, pp. 20–28. Springer, 2012.

[He] Hex-Rays SA: IDA – The Interactive Disassembler. https://www.hex-rays.com/products/ida/overview.shtml (Accessed on December 12, 2015).

[Hi15] Hill, K.: Fitbit data just undermined a woman's rape claim. http://fusion.net/story/158292/fitbit-data-just-undermined-a-womans-rape-claim/ (Accessed on November 21, 2015), 2015.

[In15] Inc., TechInsights: Deep Dive Teardown of the Withings Activité Pop HWA01 - Bluetooth 4.0 Fitness Watch. http://www.techinsights.com/reports-and-subscriptions/open-market-reports/Report-Profile/?ReportKey=10708 (Accessed on November 24, 2015), 2015.

[Kr15] Kramer, C.: Reverse-Engineering iOS Apps: Hacking on Lyft. https://realm.io/news/conrad-kramer-reverse-engineering-ios-apps-lyft/ (Accessed on November 26, 2015), 2015.

[KTK14] Kawamoto, K.; Tanaka, T.; Kuriyama, H.: Your Activity Tracker Knows when You Quit Smoking. ACM, New York, NY, USA, pp. 107–110, 2014.

[MB15] Mattu, S.; Brain, T.: Unfit Bits. http://www.unfitbits.com (Accessed on November 23, 2015), 2015.

[Ol14] Olson, P.: Fitbit Data Now Being Used In The Courtroom. http://www.forbes.com/sites/parmyolson/2014/11/16/fitbit-data-court-room-personal-injury-claim/ (Accessed on November 21, 2015), 2014.

[PI14] Paul, G.; Irvine, J.: Privacy Implications of Wearable Health Devices. In: Proceedings of the 7th International Conference on Security of Information and Networks. SIN '14, ACM, New York, NY, USA, pp. 117–121, 2014.

[RCB13] Rahman, M.; Carbunar, B.; Banik, M.: Fit and Vulnerable: Attacks and Defenses for a Health Monitoring Device. ArXiv e-prints, 2013.

[RCT14] Rahman, M.; Carbunar, B.; Topkara, U.: SensCrypt: A Secure Protocol for Managing Low Power Fitness Trackers. In: Network Protocols (ICNP), 2014 IEEE 22nd International Conference on. pp. 191–196, 2014.

[Ri12] Rizzo, J.: Hacking an Android Banking Application. http://blog.ioactive.com/2012/11/hacking-android-banking-application.html (Accessed on November 26, 2015), 2012.

[Se15] Semiconductor, Nordic: nRF51822. https://www.nordicsemi.com/eng/Products/Bluetooth-Smart-Bluetooth-low-energy/nRF51822 (Accessed on November 24, 2015), 2015.

[Sh15] Shoshitaishvili, Y.; Wang, R.; Hauser, C.; Kruegel, C.; Vigna, G.: Firmalice - Automatic Detection of Authentication Bypass Vulnerabilities in Binary Firmware. 2015.

[Te03] Telerik: Fiddler – The free web debugging proxy for any browser, system or platform. http://www.telerik.com/fiddler (Accessed on November 22, 2015), 2003.

[TKS13] Tsubouchi, K.; Kawajiri, R.; Shimosaka, M.: Working-relationship detection from fitbit sensor data. In: Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication. ACM, pp. 115–118, 2013.

[We14] Wei, J.: How Wearables Intersect with the Cloud and the Internet of Things : Considerations for the developers of wearables. Consumer Electronics Magazine, IEEE, 3(3):53–56, 2014.

[WWW15] W., Shuai; W., Pei; W., Dinghao: Reassembleable Disassembling. In: 24th USENIX Security Symposium (USENIX Security 15). USENIX Association, Washington, D.C., pp. 627–642, August 2015.

[ZP14] Zhou, W.; Piramuthu, S.: Security/privacy of wearable fitness tracking IoT devices. In: Information Systems and Technologies (CISTI), 2014 9th Iberian Conference on. IEEE, pp. 1–5, 2014.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 45

Automotive Ethernet: Security opportunity or challenge?

Christopher Corbett[1,2], Elmar Schoch[1], Frank Kargl[2], Preussner Felix[1]

Abstract: The automotive industry's future trends, such as automated driving or advanced driver assistance, require large bandwidths to handle massive data streams and strongly depend on well timed communication. The Ethernet technology is seen as a suitable candidate to cover those needs for vehicle-internal networks; however, Ethernet involves security issues. Thus, by discussing automotive Ethernet attributes with regard to the adaption of existing security mechanisms in contrast to the potential of creating new ones, several challenges and opportunities emerge in consideration of comparatively fewer available resources and the integration into a vehicle environment. Based on these results we derive and propose ideas for manipulation and misuse detection mechanisms.

Keywords: automotive, Ethernet, security, discussion, network, detection, misuse, misbehavior

1 Introduction

Modern automobiles provide drivers with a wide range of safety, comfort and assistance functionalities. The evolution of existing and the integration of additional features in new vehicle generations result in highly complex automotive systems with enhanced hardware and software parts. These systems attract the attention of computer security researchers who challenge themselves to find and exploit flaws in software and hardware implementations. Their goal is to modify, extend, tune or misuse in-vehicle devices beyond their intended purposes or restrictions [VM15]. Any modification can influence the safety state of a vehicle. Hence, it is important to protect an automotive system from any manipulation by third parties through the integration and continuous improvement of security mechanisms.

Future trends like automated driving and advanced driver assistance are seen as key features for further automotive development. They rely on increasing computing power as well as massive data exchange between sensors, actuators and processing devices. The integration of Ethernet into the automotive domain holds opportunities for new or modified security mechanisms to prevent malicious intervention, which could threaten the safety state of a vehicle.

In this paper we discuss the characteristics of automotive and common Ethernet networks within defined criteria and present their differences and potentials for security mechanisms, with focus on vehicle-internal wired networks.

[1] Audi AG, 85045 Ingolstadt, <forename> . <surname>@audi.de
[2] University of Ulm, Institute of Distributed Systems, Albert-Einstein-Allee 11, 89081 Ulm, <forename> . <surname>@uni-ulm.de

2 Criteria for discussion

In reference to the Open System Interconnection (OSI) model, we mainly focus on the physical link, data link, network and transport layers with their protocols Ethernet, Internet Protocol Version 4 (IPv4), Internet Protocol Version 6 (IPv6), Transfer Control Protocol (TCP) and User Datagram Protocol (UDP). We also consider environmental influences and impacts on vehicle parameters as appropriate factors, as the automotive network is part of a large cyber physical system. Regarding the stated premises, the list of criteria shown in Figure 1 will be the base for our discussion of security challenges and opportunities. We assume that the principles of common Ethernet networks and the used protocols are known by the reader, and we only give an introduction to automotive specifics.

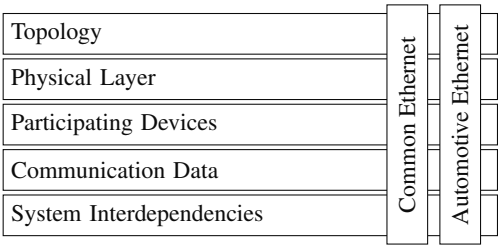
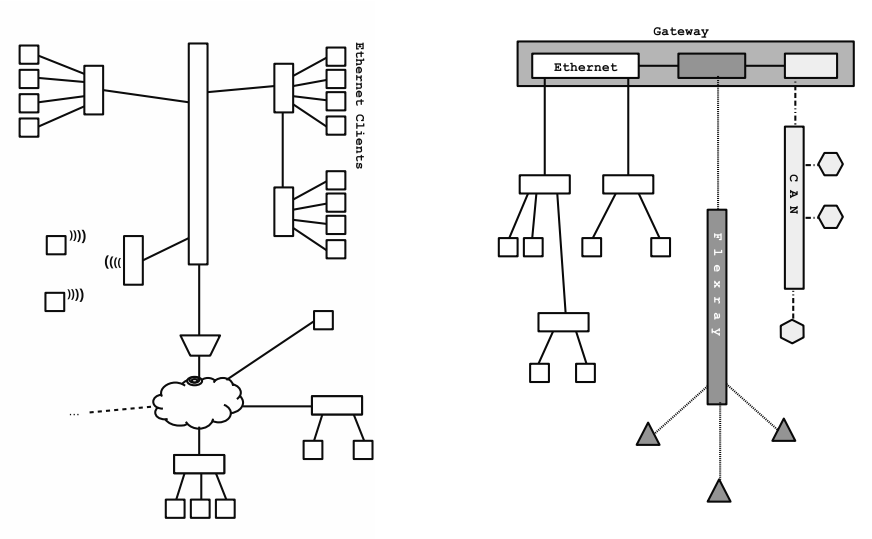


Fig. 1: List of criteria

3 Discussion

3.1 Topology

The design of automotive bus network topologies is mainly driven by functional requirements and cost efficiency. Over the past decades, several network technologies were developed (e.g. Controller Area Network (CAN), Local Interconnect Network (LIN), FlexRay, and Media Oriented Systems Transport (MOST)) in order to satisfy requirements such as time accuracy, low cost, or extensibility. This trend led to very heterogeneous internal networks where protocol conversion between different protocols becomes necessary to enable network-wide data exchange. All of these technologies are either based on arbitration or time slot mechanisms for bus access. The integration of Ethernet adds a packet and point-to-point based technology to the system. Switching and routing components are necessary to enable Ethernet, IPv4 or IPv6 communication between more than two Ethernet devices. Figure 2b shows an abstract example of an heterogeneous automotive network topology with several different bus technologies (CAN, FlexRay, Ethernet) and protocol conversion, via a central device providing gateway functionalities, in contrast to a common network topology shown in Figure 2a. Due to the functional approach, the topology is hierarchically structured, divided into several logical segments and overall considered to be highly static and not extensible.



(a) Abstract example of a common Ethernet network topology

(b) Abstract example of a heterogeneous automotive network topology

Fig. 2: Topology examples

Discussion
One of the primary differences between automotive and common Ethernet networks are the hierarchical structure and the combination of different bus network topologies. With the introduction of Ethernet to vehicle-internal networks, legacy bus technologies will continue to be present and not completely replaced. The easy extensibility and dynamic configuration of common networks must deliberately be avoided to increase controllability and create a solid foundation for safety related requirements. Thus, deliberate limitations of communication are inevitable, and must be applied for example through the use of firewalls, packet filters, network segmentations or Virtual Local Area Networks (VLAN). Major challenges for automotive Ethernet networks are to master the wide functional range of network protocols, as well as to restrict access to configurations of switching and routing components. Furthermore, changes to the topology after the engineering process must be prevented.

3.2 Physical Layer

The physical layer is responsible for transporting digital information via analog electrical signals. The selection of an appropriate transport media strongly depends on physical limitations as well as communication and application requirements. Common demands are bandwidth, unidirectional or bidirectional communication, resistance against electromagnetic influence, cost efficiency, distance to overcome and compatibility.

Modern cars process huge amounts of data provided by many sensors and actuators. Especially image processing units require high bandwidths considering increasing resolutions and a growing number of devices. Data rates of at least 100Mbit/s are necessary to satisfy those needs. Whereas applications with lower bandwidth requirements can be supported by bus technologies like LIN, CAN and FlexRay.

The development of accurate automotive Ethernet technologies is additionally driven by flexible installation and power consumption. Fiber optical cables, which are also used in MOST networks, are not as practicable in comparison to copper based cables. Currently, unshielded twisted pair copper conductors are favored for 100Mbit/s and 1000Mbit/s connections; however, due to different physical characteristics, coaxial copper cables could also be applied for the latter.

The most promising technology at the moment for fast Ethernet transceivers (100Mbit/s) is BroadR-Reach® introduced by BroadCom and standardized by the OPEN Alliance Special Interest Group (SIG). With this technology,a pair of twisted unshielded copper conductors can exchange information bidirectional over a guaranteed distance of 15m. In order to provide a bandwidth of 1000Mbit/s over the same type of wiring, 1000Base-T1 is currently the considered standard in automotive Ethernet networks. Being work in progress and because of physical restrictions between the two standards, the bandwidth of Ethernet connections is currently not negotiable at runtime and bound to the used transceivers. Transceivers of the same brand are used, in order to reduce costs and thus maximize their quantity. It is foreseeable that future transceivers will support several standards to increase flexibility in systems engineering by being able to switch between data rates.

Discussion
Cost efficiency and strong requirements on electro magnetic resistance in automotive networks limit the range of possible physical layer technologies compared to common networks. The transfer rates between two connection points are not yet negotiable and are bound to the used transceiver, as well as the used copper conductors. All physical layer attributes are known at the end of the engineering process and will not be changed during its lifetime.

3.3 Network Participants

Participants in any automotive bus network are generally called Electronic Control Units (ECU). A unit is responsible for a certain kind of functionality and specifically designed for that purpose. Environmental influences such as temperature ranges between –40°C and +100°C, humidity or vibrations affect the selection of appropriate components. Peripheral devices must meet all of the specifications and the range of possible parts is limited. Software and hardware components must meet legal and safety requirements and manufacturers are constrained to prove their compliance. Each software part, such as Ethernet stacks, operating systems and application frameworks, are specifically chosen and analyzed in detail during the development process. Some examples for deployed operating systems and frameworks are Qnx, AUTOSAR, VxWorks and Embedded Linux. Figure 3 shows the

different layers of components which can be randomly selected, but have dependencies to each other.

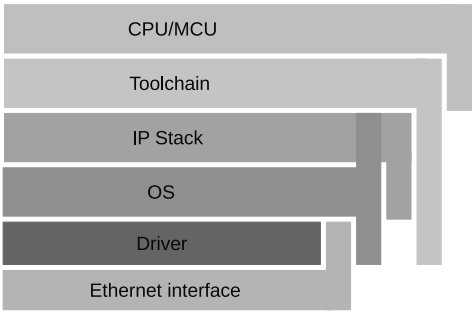


Fig. 3: Dependency overview of hard and software components

The communication within the network is designed, specified, tested and documented for every specific use case, and at the end of the development process knowledge about all network participants exists in detail and will not change after the release of the vehicle.

Discussion
For automotive Ethernet networks it is mandatory that participating devices are reliable, controllable and act strictly according to specification. The misuse of devices, such as modifications of software parts or the replacement of components with third party devices, can lead to undefined behavior and influence the safety of the vehicle. Hence, it is important to ensure the network's consistency by preventing access of unknown network participants, misbehavior of existent devices, as well as any spoofing attempts.

Yet there are no standardized mechanisms for automotive Ethernet networks to restrict access of untrusted devices. The adaption of existing authentication mechanisms (e.g. IEEE 802.1x) to automotive Ethernet networks could be a reasonable approach.

3.4 Communication data

Communication within Ethernet networks is defined by the protocol header, the information in the payload field and their interpretation. In consideration of all possible attributes we think that data structures, message frequencies and the ratio of TCP and UDP connections are promising distinctive parameters for the development of security mechanisms for automotive Ethernet networks. Therefore we focus on these attributes within this section.

Automotive systems must comply with a wide range of safety and reliability specifications. For example, the used protocols and data structures are well chosen, known in every detail and each possible impact on the whole system is considered before the end of the engineering process.

Being a cyber physical system, most of the functionality relies on control loops where sensors, actuators and processing units by design provide or consume data frequently. Events

triggered by user interactions can either result in a single and terminating task or a periodic sequence of tasks, each implicated with data exchange. Considering the use of bandwidth, memory consumption, timeout management and timing for data exchange between one or several participants, UDP connections have a smaller footprint than TCP connections, but limitations in payload size and reliability. A choice strongly depends on the requirements of each application or service and their safety impact. Yet there are no studies on the distribution of TCP and UDP connections in an automotive Ethernet network, and only estimations can be made.
The operability of most functions and eventually the safety state of a vehicle strongly depend on accurate timing, data correctness and authenticity of exchanged information between sensors, actuators and ECUs. Thus protection against manipulation takes priority over preventing eavesdropping of network data. After the market introduction of a vehicle a change or update on used protocols are very unlikely. The domain of information and entertainment (infotainment) systems can be an exception due to the interaction with consumer electronics and common Ethernet network services.

Discussion
Communication in vehicle-internal networks rests upon specifications, and once released changes are unlikely. Compared to common networks, the user of a vehicle is limited to triggering machine-to-machine interactions (e.g. data exchange, remote function calls). A user is supposed to only change predefined payload values via existing Human Machine Interfaces (HMI) (e.g. steering wheel buttons, infotainment controls), but can not alter existing protocol structures or introduce protocols of their own.
The exchange rates of information are rather high. Complete data encryption, to prevent eavesdropping and guaranty message authenticity, would cause heavy use of resources and can lead to large latencies which contradicts with timing requirements. Applying integrity and authentication mechanisms, such as Message Authentication Codes (MAC), could be a more efficient approach for such use cases.

3.5 System interdependencies

As a cyber physical system, a vehicle is exposed to different environmental factors (e.g. physical forces, traffic participants, temperature changes) with impacts on the driver, sensors, actuators and finally the overall system state. Actions taken by the driver result in transitions between different system states and can lead to changes of environmental factors, which again influence inputs to control loops. The Ethernet network and all of its devices, being part of the vehicle, are also influenced by system state transitions.

Discussion
Devices in automotive networks are not geographically spread and are clustered in a superior system, i.e. a vehicle, controlled by a driver and partly by passengers. Compared to common networks, protocols and data structures are strictly specified and user interaction via control interfaces explicitly limited. Environmental factors and user actions trigger dedicated functions and change the vehicle system state as well as the exchanged data streams.

4 Security challenges and opportunities

Based on the previous discussion, we derive numerous challenges and opportunities for automotive Ethernet networks and briefly describe them in the following section.

4.1 Challenges

Increasing number of potential attackers
Ethernet is a well-known technology, which is extensively described in literature and many people have experience with this technology through day-to-day use both at work and at home. Therefore the number of capable potential attackers is considered greater than compared to other automotive bus technologies. Due to the availability of cheap compliant devices, the technical barrier for interested persons is also rather low. In combination with todays social media platforms (e.g. Youtube, Forums) it is possible to quickly share information and provide simple instructions for carrying out exploits.

Fewer resources
In comparison to systems using common Ethernet, resources (e.g. computing power, memory) are more restricted due to financial and vehicle power boundaries. As such it is not possible to run full-fledged security appliances, such as Intrusion Detection Systems (IDS) or high performance network traffic inspection.

Gained attention
Ethernet will be the key technology for future automotive trends to fulfill necessary requirements in vehicle-internal networks. Being a daily used object, and with a total of 44.4 million registered vehicles [Kr15] alone in Germany, cars are commercially attractive to numerous interest groups. We defined a list of stakeholders, shown in table 1, with their main intentions on exploiting software and hardware flaws.

Tuner/Modder	**Criminals**	**Third party manufacturers**	**Hackers**
- Modifying funct.	- Theft	- Providing alteration	- Sell Knowledge
- Extending funct.	- Selling used parts	- Provide spare parts	- Personal satisfaction
- Tuning of components	- Life threatening manipulation		- Searching for flaws

Tab. 1: List of interest groups for exploits

Accessibility
All network components are clustered inside a vehicle and not physically distributed among different geographical locations, like in most common networks. Who ever is in control of the automobile has full access to the whole network ecosystem.

Security diversity
An automobile is a very heterogeneous system. Each distributed communication technology was designed for a specified purpose, provides certain features and has furthermore limitations. It is difficult to apply a mutual security mechanism to the system, e.g. MAC, due to differences in computing resources of ECUs and payload sizes of protocol frames (e.g. CAN (8 Byte), Controller Area Network with Flexible Datarate (CAN-FD) (64 Byte), FlexRay (255 Byte) and Ethernet (1500 Byte)).

Safety compliance
Safety related functions in automotive systems must meet certain requirements, such as high availability, strong reliability, exact timing and low latency in communication. Hence, it is important that applied security mechanisms do not have impact on the safety state of a vehicle, as safety is prior to security in vehicular systems.

4.2 Opportunities

Detection capabilities
Attributes of automotive networks, regarding running applications, services, architecture and topology design, are very static. The combination of this static design with environmental factors and vehicle states enables further detection capabilities of contrary behavior in Ethernet networks after the manufacturing process.

Extended bandwidth
With the introduction of Ethernet to the vehicle-internal networks the bandwidth compared to legacy bus systems is increased and gives leeway to better authentication or encryption mechanisms (e.g. Media Access Control (MAC)). Also software or configuration updates can be distributed faster.

Stricter separation
Due to the point-to-point characteristics of Ethernet a stricter separation into and within functional domains can be achieved compared to bus technologies like CAN. Through the use of switches in combination with the IEEE 802.1Q specification of VLAN and Quality of Service (QoS), as well as firewall concepts, communication paths inside the network can be restricted and priorities for packets considered.

4.3 Approaches for security mechanisms

The previously stated challenges and opportunities hold potential for different approaches to security mechanisms for automotive Ethernet networks. In this section we propose our ideas regarding applicable mechanisms with focus on the detection of network manipulation and misuse.

Physical Layer security mechanism
Security mechanisms for common networks based on layer 1 attributes are not reasonable due to a large variety of available transceivers, cable types, range extenders and negotiable transfer rates. Through the massive restrictions in automotive Ethernet networks we think of physical layer security mechanisms to verify the identity of physically to each other connected devices by searching for distinct characteristics as electric resistance, impedance or changes of signal modulations, frequencies and amplitudes to build a distinct fingerprint. Also the introduction of a light-weight encryption mechanism could be possible to prevent the addition of new or the replacement of existing devices.

Meta information based mechanism
Through the restricted design of ECUs and the detailed knowledge about all network related parameters and attributes, meta information (e.g. stack auditing, packet statistics, used parameters) could be used to detect misbehavior or spoofing of devices in vehicle-internal networks.

Specification based misuse detection mechanism
Considering the stated communication data characteristics the introduction of an anomalous behavior detection system (or Misuse Detection System (MDS)) based on specifications could be used to uncover possible manipulations or injection of malicious network data.

Misbehavior detection mechanism through vehicle state mapping
By mapping certain identified system states to exchanged data streams, atypical behavior either by the system or the network could be detected.

5 Conclusion

In this paper we briefly discussed common and automotive Ethernet networks within defined criteria to identify challenges and opportunities for security mechanisms in vehicle-internal networks. Overall we see potential for network manipulation and misuse detection mechanisms in future automotive Ethernet networks. We consider the combination of several mechanisms as a chance to increase the security of future vehicular Ethernet networks, with a sensible use of resources. The elaboration, simulation and evaluation of all ideas are topics for future research.

References

[Co14] Corporation, Broadcom: . BroadR-Reach Physical Layer Transceiver Specification for Automotive Applications., May 2014.

[Kr15] Kraftfahrt Bundesamt: , Gesamtbestand an PKW zum 1. Januar 2015 in Deutschland, January 2015.

[PH12] Peter Hank, Thomas Suermann, Steffen Mueller: Automotive Ethernet, a Holistic Approach for a Next Generation In-Vehicle Networking Standard. Advanced Microsystems for Automotive Applications, pp. 79–89, 2012.

[Ta96] Tanenbaum, Andrew S.: Computer Networks. 1996.

[VM15] Valasek, Chris; Miller, Charlie: Remote Exploitation of an Unaltered Passenger Vehicle. Black Hat Conference, 2015.

[We05] Werner, Martin: Netze, Protokolle, Schnittstellen und Nachrichtenverkehr. 2005.

[Wi13] Wilfried, Plassmann: Handbuch Elektrotechnik. Springer Vieweg, 6 edition, 2013.

[Zi07] Zimmermann, Schmidgall: Bussysteme in der Fahrzeugtechnik. Vieweg, 2nd edition, 2007.

[ZMC09] Zhang Min, Dusi Maurizio, John Wolfgang; Changjia, Chen: Analysis of UDP Traffic Usage on Internet Backbone Links. Applications and the Internet, 2009. SAINT '09. Ninth Annual International Symposium on, 2009.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 55

A Semantic Framework for a better Understanding, Investigation and Prevention of Organized Financial Crime

Ronny Merkel[1], Christian Krätzer[1], Mario Hildebrandt[1],
Stefan Kiltz[1], Sven Kuhlmann[1], Jana Dittmann[1]

Abstract: Using semantic technology for data storage and exploration is an important issue in computer science, however barely applied to forensic investigations. In this paper, a conceptual framework is proposed for the detailed modeling of structured domain knowledge in the field of organized financial crime, with a special focus on sparse information (e.g. flows of money, data and know-how, exploited vulnerabilities and attackers motivation) and the proposition of a credibility measure (to rate the reliability of used information based on open source intelligence, expert surveys and captive interviews). In addition to the ontology-based, abstract domain knowledge model, the proposed framework consists of an explorative information discovery functionality, which can couple concrete, case-related data from different knowledge bases with the abstract domain knowledge, to assist experts in the investigation of crimes and the discovery of new relations between different pieces of evidence. The proposed framework is illustrated using the exemplary use case scenario of Point-of-Sale (POS) Skimming. Furthermore, its flexibility, scalability and a potential integration into current and emerging police standards is discussed.

Keywords: semantic modeling, organized financial crime, ontology, sparse information, credibility measure, explorative search

1 Introduction

Using semantic technologies is an important issue in computer science. Originating in linguistics and the philosophy of language, the modeling of meaning (semantics) is related to how we think (process information) and talk (express information), which can also be applied to digital data. With the idea of modeling semantics in the Web, Tim Berners-Lee has initiated a decade of research into this topic, utilizing semantic models for an increased accuracy, speed and flexibility of data processing [BHL01]. Exemplary use cases for semantic models include the fusion of information from different data storages, the extraction and structuring of information from unstructured data as well as the presentation of structured data for human processing, e.g. for an exploratory search.

In the scope of this paper, semantic technology is discussed from a forensic perspective. The idea of using semantics for forensic casework is not new and has been proposed in

[1] Workgroup on Multimedia & Security, Otto-von-Guericke University of Magdeburg, Universitaetsplatz 2, 39106 Magdeburg, {merkel, kraetzer, hildebrandt, kuhlmann, kiltz, dittmann}@ovgu.de

The work in this paper has been funded in part by the German Federal Ministry of Education and Research (BMBF), funding code 13N13473.

literature as well as in proprietary software tools. However, when looking at specific casework examples, it seems that these technologies are barely used in daily police work. The aim of this paper is therefore to propose a conceptual framework for modeling semantic information in the forensic context, exemplary illustrated using an organized financial crime use case. In particular, the contribution of this paper is as follows:

- A conceptual framework is proposed for modeling and usage of semantic information in forensic investigations, based on the exemplary field of organized financial crimes. The model includes a manually created ontology (representing structured, abstract domain knowledge), case-specific information (e.g. from concrete knowledge bases) and intermediate interfaces usable by forensic experts for abstract information modeling and explorative information discovery (views).

- When designing the framework, a specific focus is laid on three major issues. During the ontology creation, a specific focus is on the modeling of sparse information (e.g. money, data and know-how flows, vulnerabilities and motivation). A source-dependent credibility measure is proposed to assure the quality of the model (based on open source intelligence, expert surveys and captive interviews). Also, a forensic investigators view is designed to allow for explorative information discovery.

- The practical application of the scheme is illustrated using the example of Point-of-Sale (POS) Skimming from the domain of organized financial crimes. The procedure is demonstrated showing exemplary aspects of the ontology modeling, the explorative search of forensic experts as well as the knowledge base updating.

- Flexibility and scalability as well as a potential integration into contemporary police data exchange standards (such as FIDEX [Lot10] and XPolizei [Haa11]) are suggested and discussed, to provide easy extensibility of the framework.

The authors acknowledge the nature of this paper as work in progress. However, an early inclusion of the scientific community is regarded as a vital prerequisite to assure good scientific practice, especially in respect to the Daubert criteria for acceptability in court, such as scientific peer review, general acceptance and error rates [Dau13]. The remainder of this paper is structured as follows: section 2 gives a brief overview over relevant state of the art, followed by an introduction of the proposed framework in section 3. The three issues of major focus are discusses in section 4, section 5 illustrates the framework for a POS Skimming use case. Flexibility, scalability and integration into police standards are discussed in section 6. Future research is highlighted in section 7.

2 State of the Art

On a syntactical level, **taxonomies** can be considered as a form of hierarchical categorization of certain domain knowledge, often visualized in the form of a tree, e.g. for content management or information retrieval [Pfu12]. However, to also model complex

relations between different entities, computer scientists have adapted the concept of **ontologies** from Philosophy (the study of being and the existence of entities as well as their conditions and principles) [GI02]. Ontologies are used as a formal means of modeling semantic information of a certain knowledge domain in a structured and standardized way for computer communication, automatic inference, data representation and mining [GI02]. Today, semantic technologies are subject to broad research activities and a wide range of applications. The numerous protocols are summarized under the semantic web stack [W3C15].

In **forensic investigations**, machine-learning techniques are applied for detecting crime pattern, e.g. in [WRW+13]. Semantic models seem to be used in very selective cases only. Early semantic models were applied to synthesize alibis, e.g. by analyzing texts [Nis12]. A few examples of more recent approaches include proprietary software for police investigations (e.g. [Tex15] for semantic text mining, [Jac12] for searching structured and unstructured data as well as [Mar15] for tagging, searching and data integration). However, these are proprietary approaches and no information is openly available about their specific concepts, realization and performance. The possibilities of semantic technologies for police work and forensic investigations are partly discussed in [Liu12] and [HWS15]. A more comprehensive approach using semantic modeling with the help of an ontology is proposed in [ASB+14], based on structured and unstructured data from different police databases and freely available texts for the recognition of crime threats. It is regarded here as the first proposal for a practical realization of semantic approaches for forensic casework, allowing for data search, linking, exploration, modeling and visualization. However, the work is introduced on a very abstract level and only few specific details on the modeled components are given, e.g. the concrete entities of the underlying ontology. Especially the reliability of the (often automatically) extracted entities is unclear in the approach, which is described from a rather top level view. In comparison, the **here presented work** starts from a specific application scenario, especially focusing on sparse information and a credibility measure to assure the relevance and reliability of the extracted information. Scalability, flexibility and integration issues of the scheme are then discussed, making it a bottom up approach.

3 A Conceptual Framework for Semantic Information in Forensics

In forensic investigations, information is usually provided on different levels of abstractness: case-related information is usually concrete, such as the time and place of the crime, the damage caused or the number of people involved. Abstract information includes the experience of the forensic expert, such as typical modi operandi, places where traces can typically be found or common procedures of investigation. The expert has to combine these different levels to gain information. Using semantic modeling, the storage and processing of data from these different levels of abstractness can be facilitated through digital means, assisting the forensic expert in the investigation (Fig. 1). On the **abstract knowledge layer**, the overall experience of the forensic expert might be ex-

tended by information obtained from convicted criminals and open source intelligence and can be stored in the form of an ontology (left image). Such *ontology* (right image) contains no specific instances and represents the abstract domain knowledge of the application scenario, structured in a machine-readable form. It includes a hierarchical class structure with classes (e.g. 'attacker', 'vulnerability', 'attacking tool' or 'defense mechanism'), subclasses (e.g. 'human-based', 'technical' or 'organizational' 'vulnerabilities'), class attributes (e.g. 'credibility measure' providing reliability information, 'affected security aspects' like confidentiality / integrity or 'error rates' of an investigation method) and relations between different classes (e.g. a certain 'attacker' might use a certain 'attack tool', an 'attack' might exploit a certain 'vulnerability'). Two different types of ontologies are proposed in this work: a general *crime field ontology* to represent general domain knowledge as well as *additional ontologies*, representing sub-field specific knowledge to be integrated (e.g. knowledge about forensic traces found at a crime scene, like fingerprints or toolmarks). Additional ontologies can be integrated into the crime field ontology using hooks (e.g. a class 'trace' might be hooked up with an additional ontology 'fingerprints', which holds the domain knowledge of forensic fingerprint investigations). All ontologies are controlled using an *ontology manager* (*additional ontology manager* respectively). Such manager consists of a *user level* taking *request parameters* (e.g. creation / deletion of an ontology, reading global or local contexts represented by the relational neighborhood of a class, appending, renaming, deleting or moving of classes, change of attributes or relations). A requested operation is checked at the *system level* concerning its validity and then converted to appropriate read, write or delete commands, which are then applied to the specific ontology. Request parameters for reading, writing and deleting can be issued by the *ontology designer view*, used for manual update of the ontology (only option of gaining write access to the semantic domain model). Because the relationships between different entities of the ontology can be very complex and small changes might lead to a significant amount of remodeling of dependent relations (and might require updates of the database structure of the concrete knowledge layer), only an expert is allowed to manually perform this task.

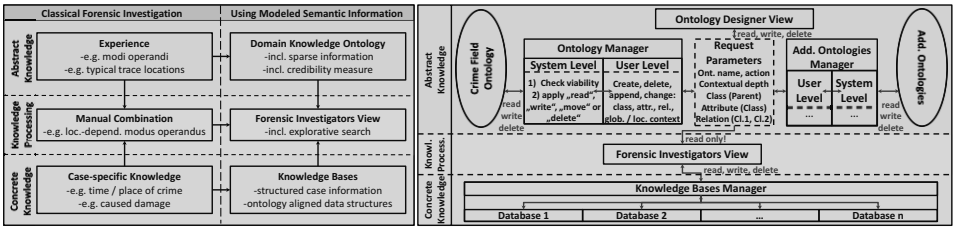


Fig. 1: Forensic abstraction layers: abstract knowledge (blue), knowledge processing (grey), concrete knowledge (green). Left: forensic investigations without and with the help of the proposed semantic modeling framework. Right: proposed interface structure.

On the **knowledge processing layer**, the *forensic investigators view* provides the main semantic functionality required for assisting the forensic expert in the investigation (left image). The investigator selects a relevant entity to explore (class of the crime field

ontology from a presented list view) as well as a contextual depth, which form the request parameters sent to the ontology managers of the respective ontologies. Only reading access is allowed to ensure the consistence of the abstract knowledge relations. The local (or global) context received by the forensic investigators view is forwarded to the *knowledge bases manager* in the form of a sub-ontology, where it is filled with all specific instances available in the *databases* for the classes provided and returned to the forensic investigators view (right image). Here it is displayed, showing the network of relations between the entities of the selected neighborhood as well as the associated instances (see also section 5). Thus, the investigator can explore important relations (e.g. which 'attack tool' has been used by a certain 'attacker'), potentially additional relations (e.g. other 'crimes' committed by the 'attacker') as well as all similar cases (e.g. all 'crimes' in which a certain 'attack tool' was used). Furthermore, the forensic expert can input new case-related information (specific instances of the selected local context), which can be directly stored in the corresponding databases of the knowledge bases manager. The forensic investigators view is depicted for a specific use case in section 5.

On the **concrete knowledge layer**, case-specific knowledge is usually available in the form of digital or physical files and folders in an unstructured form or in structured databases (left image). To allow for a reliable processing of this information, the *knowledge bases manager* needs to structure and store this data in respective databases (right image). Data might be structured automatically using certain data mining techniques (e.g. text mining). In the here presented scheme, a manual structuring of data is proposed for higher reliability. Data might furthermore be structured in two possible ways. In a **first general data structuring effort**, the global context of a complete ontology might be requested from the *ontology manager* of the *abstract knowledge level*. This is a read only access, but provides all class entities of an ontology. Based on these entities, database schema can be arranged in a way to store data using structures similar to the ontology, e.g. similarly named database keys. Specific case-related datasets then need to be manually entered by forensic experts according to the provided keys. In case of changes applied to the ontology via the *ontology designer view*, which have lead to structural changes in the data already stored in the database, database restructuring rules might have to be manually defined. In a **second general data structuring effort**, additional case information might be entered by the forensic expert during case analysis, using the *forensic investigators view*, which is sent to the knowledge bases manager and stored in respect to the corresponding database keys.

4 Sparse Information, Credibility & Explorative Search

In the scope of this paper, a special focus lies on the relevance and reliability of the designed ontology, to maximize the benefit for forensic investigations. To realize this goal, three main aspects are particularly considered. **Sparse information** is often not present to forensic investigators and therefore not included in the investigation. For example, the *flow of money* after a successful crime is often not known and even if a person is con-

victed, the whereabouts of the money are often undiscovered. If typical hideouts or techniques of money laundering can be acquired from additional sources (e.g. by interviewing convicts in prisons), this information might be useful to look for signs of specific hiding or laundering techniques. Therefore, sparse information is of great importance to the issue. In the scope of this paper, five specific types of sparse information are proposed (and first findings included in the ontology of the exemplary use case introduced in section 5). The *flow of money* describes typical money storing, laundering and spending behavior, which might help forensic investigators to direct their attention towards related activities. The *flow of (stolen) data* reflects on how criminals communicate, trust levels within an organization as well as techniques to convert stolen data into money. Such information seems vital for solving crimes. The *flow of know-how* indicates ways in which knowledge is acquired (e.g. from educational institutions, the internet or insider knowledge). Identifying sources of knowledge might help forensic investigations to narrow down the set of suspects in certain cases and seems very helpful for preventive applications. *Commonly exploited vulnerabilities* explicate preferably targeted weaknesses (e.g. organizational, human or technical). Such knowledge can be used especially in preventive work, but also to direct investigations. The *motivation of an attacker* illustrates reasons why criminal actions are conducted (e.g. personal gain, blackmail), which might be extracted successfully from captive interviews, providing valuable information towards the prevention of future crimes. To include such information into the modeling process, the combination of open source intelligence, expert knowledge and captive interviews seems very promising.

To measure the reliability of modeled semantic information, a **credibility measure** is proposed. The measure is stored in the form of ontology attributes (section 3), which are saved as a three-tuple for each class of the complete ontology:

$$\text{Credibility measure: } C(\text{OSINT},\text{ES},\text{CI}) \quad \{\text{OSINT}, \text{ES}, \text{CI}\} \in \mathbb{N}, 0 \leq \{\text{OSINT}, \text{ES}, \text{CI}\} \tag{1}$$

The tuple of formula (1) contains the amount of (independent) *open source intelligence sources* (OSINT), *surveyed experts* (ES) and *captives interviewed* (CI). Sources are considered independent if they are from different online sources, experts of different forensic departments or captives from different jails. It is acknowledged that such classification as well as the information provided by a human is subjective in a particular way. Objectivity is therefore aimed at by regarding only these entities as credible, which are acquired by a certain amount of different individuals, preferably from different origins (e.g. forensic experts vs. captives). An interesting aspect of future research can be seen in answering the question of how to deal with conflicting or opposing information from different sources. These cases should also be incorporated in a comprehensive credibility measure in the future.

An **explorative search view** is proposed as a main feature of the framework. Criminal investigators require the structured visualization of relevant entities and their relations and at the same time a reduction of irrelevant information. With keyword-based search engines, this goal seems not achievable. Displaying manual selected objects of interest and a flexible visualization of their affiliated objects, relations and available concrete

instances in varying depths seems a potential approach towards this challenge. Furthermore, the investigator has the option of storing additional case-related information during his analysis, which will be automatically saved by the knowledge bases manager. At the same time, the investigator is protected from accidental changes of the domain knowledge, because the ontology itself is write-protected.

5 Exemplary Use Case: Modeling Point-of-Sale (POS) Skimming

Point-of-Sale (POS) skimming is a well-suited example to visualize the different modeling steps of the proposed scheme. During such offense, a criminal gains access to a POS terminal (e.g. in a supermarket without cameras) at night, installs a hidden skimmer and escapes undetected. During normal operations of the supermarket, the magnetic stripe of credit or debit cards of customers is captured automatically by the skimmer, encrypted and sent to the criminals. The datasets are then decrypted by the technician of the criminal organization and stored onto fake cards used for shopping or funds withdrawal.

To **model the semantic domain knowledge** of such incident, several classes might be defined. For example, a 'criminal organization' might consist of different 'attackers', performing a certain 'offense' (e.g. POS skimming). They use specific 'attack tools' (e.g. skimmer), which exploit certain 'vulnerabilities' (in this case a lack of cameras in the supermarket or insufficient access restriction measures). The customers can be considered as 'attack target' and the attack causes a certain 'data flow' (stolen credit card data) and 'money flow' (stolen money). Earlier, a 'know-how flow' had taken place, transferring the crime-relevant knowledge to the criminal organization (e.g. from social engineering, or open source knowledge from universities or the internet). At some point, there might be a 'crime discovery' (e.g. reports of unauthorized withdrawals of funds). Forensic 'investigators' start applying 'investigative procedures' (e.g. tracing card data back to the supermarket, checking the POS terminals, etc.). The investigations can lead to certain 'consequences' (e.g. replacement of terminals, arrest of identified criminals) or additional 'defense mechanisms' (e.g. improved tamper protection of POS terminals, cameras inside the supermarket, better access restriction). For each of these classes, numerous subclasses might be modeled. For example, a 'money flow' might be further divided into the 'type of money' involved, its 'sender' and 'receiver' as well as the 'type of transfer'. In case of POS skimming, a 'money flow' might refer to type 'cash', transferred between a 'victim' and a 'criminal' via 'withdrawal'. A small section of the class structure is exemplary visualized in Fig. 2 (left) using the modeling language OWL2 [OWL12] and the open source toolkit NeOn [NeO14] (which can be considered as the ontology designer view in this example). Considering the credibility measure designed in section 4, each class of the hierarchy can only be considered as credible if a certain amount of sources specified in the *C(OSINT,ES,CI)* attribute of the class confirms its relevance for practical investigations. In Fig. 2 (left), the exemplary chosen transfer type of 'withdrawal' was reported by several websites, two of which are included here for illustration purposes [Gae13], [Kel15]. Furthermore, police authorities from two different countries

have confirmed in personal interviews the withdrawal of money using fake copies of skimmed credit or debit cards. The resulting credibility measure *C* would therefore take the form *C(2,2,0)*. The captive interviews are currently ongoing and are planned to further consolidate this credibility by increasing also the *CI* value of the tuple. Apart from modeling the hierarchical structure, the ontology also consists of several relations between the class entities. For example, a criminal organization 'consists' of attackers, which 'conduct' an offense, 'working with' a certain attack tool. Exemplary relations of the use case are visualized in Fig. 2 (right).

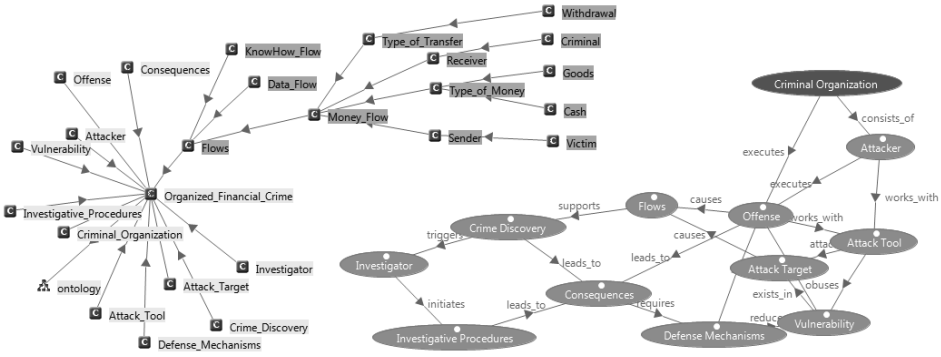


Fig. 2: Exemplary class hierarchy section for POS skimming (left) using the ontology designer view (here: NeOn toolkit [NeO14]). Exemplary visualization of selected relations (right).

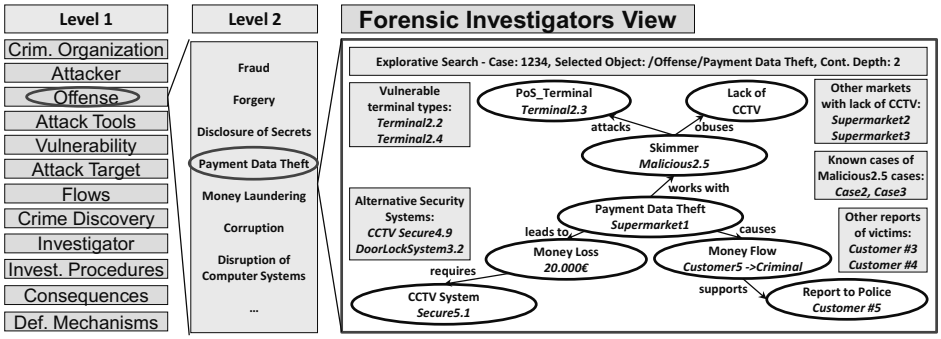


Fig. 3: Conceptual design of the forensic investigators view (POS skimming). Left: hierarchical ontology class structure from which investigated entity is chosen. Right: explorative search view.

To conduct an **explorative search** for the POS skimming scenario, the abstract ontology model is combined with concrete case information in the forensic investigators view, to assist the information extraction by an expert (Fig. 3). For that task, the class to be investigated is chosen from the hierarchical structure of the ontology (left part). In the given example, the type of 'offense' called 'Payment Data Theft' is investigated, which is located at the second hierarchy level of the ontology. The explorative investigation of the chosen class is conducted in the explorative search view (right part). A contextual dept of two is used, meaning that the neighborhood of two nodes from the 'Payment Data

Theft' is investigated in the scope of case '1234'. In this case, payment data has been stolen from 'Supermarket1'. The chosen neighborhood is sent to the knowledge base and the corresponding information of all available relevant cases is attached to the ontology classes in the form of instances. The expert can now explore these cases using the forensic investigators view. For example, he can analyze the specific skimming device applied ('Malicious2.5') and browse other cases in which the device has also been used. He could furthermore expose the terminals being (apart from 'Terminal2.3') also vulnerable to such attack (e.g. 'Terminal2.2', 'Terminal2.4') as well as supermarkets also lacking sufficient security mechanisms (e.g. 'Supermarket2', ' Supermarket3'). Regarding money loss, the expert could enquire alternative security systems to prevent this kind of attack in the future (e.g. 'CCTV Secure4.9', 'DoorLockSystem3.2') or browse other victims (e.g. 'Customer #3', 'Customer #4') for additional evidence. The view provides the information present in the knowledge base in a highly structured (semantical) way and includes only reliable (using the credibility measure) information, complemented by sparse information. It therefore exhibits the potential for a very comprehensive and resource-effective way of exploring a case and linking it to related information of other cases.

Within the different databases managed by the **knowledge bases manager**, the concrete case information of the POS skimming scenario is stored in a structured way. This means that the data is sorted according to the classes provided by the ontology. For investigating specific cases, the ontology should therefore be fixed and no changes should be applied to its structure. However, case-specific information can easily be added to the databases using the forensic investigators view. For example, if an expert has discovered that a currently analyzed skimmer is similar to 'Malicious2.5', the novel data stemming from this case (e.g. additional attacked terminals or exploited vulnerabilities) can be manually assigned to the forensic investigators view and will automatically be written to the appropriate databases by the knowledge bases manager. In case new semantic categories (ontology classes) need to be designed (e.g. as a consequence of changed modi operandi), the ontology has to be changed by an expert using the ontology designer view. If these changes create new classes, rules have to be provided for the knowledge bases manager to rearrange corresponding database keys and to map the stored entries accordingly.

A **practical validation** of the proposed scheme using large amounts of modeled entities and relations, also including specific measures of the scheme's contribution to the crime pattern extraction efforts of forensic practitioners in the field cannot be included into this paper, but remains an important issue for future work.

6 Flexibility, Scalability and Integration into Police Standards

In police work the investigation of organized crime requires a constant exchange of information across borders. This, however, is often impeded by the utilization of a multitude of commercial and incompatible software products and data formats in daily police

work. Thus, in many countries the predominant form of data exchange is via hardcopies. Especially in the context of organized crime this is slowing down case work due to the lack of a proper way of searching the documents or the possibility of applying data mining technologies to find similar patterns within the course of events. Hence, in various countries initiatives for standardized data exchange formats exist. Two examples are FIDEX [Lot10] which is relying on the National Information Exchange Model (NIEM [Nie15]) in the US and XPolizei [Haa11] as a XÖV standard in Germany. Especially regarding XPolizei, only a limited amount of information is publicly available. The XÖV standard defines process models, messages, semantic data type and code lists [BBH+13]. In contrast to that, NIEM primarily focuses on data exchange formats in the Information Exchange Package Documentation (IEPD) and a data model representing a dictionary of terms, definitions, formats and relationships [Nie15]. Thus, it primarily covers the messages and the semantic data types of the XÖV standard. The specific implementation of FIDEX defines two IEPDs: forensic case submission for the communication between forensic labs and police agencies and disposition reporting for communication with the court. Both approaches share XML as the used modeling language.

Toward the flexibility, scalability and integration into those standards no particular challenges arise since both formats specify how data should be transferred. The ontology itself can be easily integrated into the exchanged data because it utilizes XML data structures as well. However, with a growing amount of information within the ontology this might lead to an increased overhead of redundantly transmitted data. Thus, for the sake of scalability, very detailed knowledge can be outsourced into the additional ontologies (Fig. 1). It is furthermore reasonable to integrate versioning information within the ontology and to create an exchange policy for updated versions. This would only require including the version and specific items into the data exchange for a single case. With the matching ontology, the implications and relations between the items can be decoded locally. With respect to NIEM, this would also require two IEPDs, one for exchanging updated versions of the ontology (e.g. push or pull scenarios) and the exchange of the specific case work, which could be also integrated into the FIDEX Forensic Case Submission IEPD. In XÖV, different semantic data types can be defined for the exchange of ontologies. However, both approaches are limited to a national application because no global data exchange standards exist, yet. This is the primary requirement for implementing a flexible and efficient solution for a multi-national investigation of organized crime. Besides particular data standards, the compatibility of the legal requirements including data protection laws need to be considered.

7 Conclusion and Future Work

In this paper, a conceptual framework has been proposed to model structured domain knowledge from the field of organized financial crime using semantic technology, also including sparse information and a credibility measure based on open source intelligence, expert surveys and captive interviews. The framework has furthermore been ex-

tended by an explorative information discovery functionality, linking the modeled abstract domain knowledge to case-specific facts stored in knowledge bases and therefore allowing experts to explore available information in a comprehensive and efficient way. The framework has been illustrated using the exemplary use case of POS Skimming. Its flexibility, scalability and potential integration into current and emerging police standards have been discussed.

Future work should include the in-depth modeling of other modi operandi in organized financial crime, as well as a comprehensive amount of practically relevant entities and relations. Also, the possibilities of modeling time-related components should be investigated, e.g. using simulation tools. Furthermore, the inclusion of tools for mining unstructured data would enable the inclusion of additional, automatically extracted data. In this regard, also the credibility measure needs to be adapted, to allow for a reliability assessment of the extracted data and potentially opposing information from different sources. The model should be subject to a practical validation, to assess its specific contribution to forensic practitioners in finding crime patterns (including objective quality measures). Also, it might be of interest to analyze potential countermeasures of criminals towards the proposed scheme (e.g. deliberate variations of crime pattern) and the vulnerability of the approach towards such attacks. Overall, addressing these issues in future work might enable the suggested framework to be applied in practical investigative as well as preventive police work, including additional use case scenarios.

References

[BHL01] Berners-Lee, T.; Hendler, J.; Lassila, O.: The semantic web. Scientific American 284(5), pp. 28-37, 2001.

[Nis12] Nissan, E.: Computer Applications for Handling Legal Evidence, Police Investigation and Case Argumentation. Springer Science & Business Media, vol. 5, 2012.

[Tex15] Textron Systems: IMPACT. Available at: http://www.textronsystems.com/products/advanced-information/impact, last accessed: 17.11.2015.

[Mar15] MarkLogik: Semantics. Available at: http://www.marklogic.com/what-is-marklogic/features/semantics/, last accessed: 17.11.2015.

[Jac12] Jackson, R.: NZ Police deploy semantic search technology. Available at: http://www.computerworld.co.nz/article/490838/nz_police_deploy_semantic_search_technology/, last accessed: 17.11.2015.

[Liu12] Liu, J.: How can the Semantic Web help Law Enforcement? 2012. Available at: http://lib.post.ca.gov/lib-documents/CC/Class50/50-Liu.pdf, last accessed: 17.11.2015.

[HWS15] Hollywood, J.S.; Woods, D.; Silberglitt, R.; Jackson, B. A.: Using Future Internet Technologies to Strengthen Criminal Justice. 2015. Available at: http://www.rand.org/content/dam/rand/pubs/research_reports/RR900/RR928/RAND_RR928.appendix.pdf, last accessed: 17.11.2015.

[ASB+14] Adderley, R.; Seidler, P.; Badii, A.; Tiemann, M.; Neri, F.; Raffaelli, M.: Semantic mining and analysis of heterogeneous data for novel intelligence insights. Proceedings of The Fourth International Conference on Advances in Information Mining and Management, IARIA, pp. 36-40, 2014.

[Dau13] The Daubert Standard: Court Acceptance of Expert Testimony. Available at: http://www.forensicsciencesimplified.org/legal/daubert.html, last accessed: 23.11.2015.

[Pfu12] Pfuhl, M.: Taxonomien. Available at: http://www.enzyklopaedie-der-wirtschaftsinformatik.de/lexikon/daten-wissen/Wissensmanagement/Wissensmodellierung/Wissensreprasentation/Semantisches-Netz/Taxonomien, last accessed: 23.11.2015.

[Gi02] Gesellschaft für Informatik (GI): Ontologie(n). Available at: https://www.gi.de/index.php?id=647&tx_ttnews[tt_news]=57&cHash=fa4532c7f4dcda05664600a9738f3177, last accessed: 23.11.2015.

[W3C15] Berners-Lee, T.: W3C Architecture. Available at: http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html, last accessed: 23.11.2015.

[Lot10] Lothridge, K.: Forensic Information Data Exchange (FIDEX) – Final Project Report. 2010. Available at: https://www.nfstc.org/wp-content/files//FIDEX_Final-Project-Report_Final_v2_wcopyright.pdf, last accessed: 24.11.2015.

[Haa11] Voss-de Haan, P.: XPolizei - Aufbau eines einheitlichen Repository für polizeilich relevante Kerndatenobjekte. 2011. Available at: http://www.xoev.de/detail.php?gsid=bremen83.c.11268.de, last accessed: 24.11.2015.

[BBH+13] Büttner, F.; Bartels, U.; Hamann, L.; Hofrichter, O.; Kuhlmann, M.; Gogolla, M.; Rabe, L.; Steimke, F.; Rabenstein, Y.; Stosiek, A.: Model-driven standardization of public authority data interchange. Science of Computer Programming, vol. 89, part B, pp. 162-175, 2014.

[Nie15] NIEM: National Information Exchange Model. Available at: https://www.niem.gov, last accessed: 26.11.2015.

[NeO14] The NeOn Toolkit. Available at: http://neon-toolkit.org/wiki/Main_Page.html, last accessed: 07.12.2015.

[OWL12] W3C: OWL 2 Web Ontology Language Document Overview (Second Edition). Available at: http://www.w3.org/TR/owl2-overview/, last accessed: 07.12.2015.

[Gae13] Gäubote Herrenberger Zeitung: Banken sperren tausende EC-Karten. Available at: http://www.gaeubote.de/gb_10_110356796-24-_Banken-sperren-Tausende-EC-Karten-.html?GBID=766bfa500ab5af3ca062dc6e6df915d8, last accessed: 07.12.2015.

[Kel15] Kelly, N.: How White Card Fraud Works. Available at: http://www.sid.in-berlin.de/nedkelly-world/howwhitecardfraudworks.html, last accessed: 07.12.2015.

[WRW+13] Wang, T.; Rudin, C.; Wagner, D.; Sevieri, R.: Learning to detect patterns of crime. Machine Learning and Knowledge Discovery in Databases, pp. 515-530, Springer Berlin Heidelberg, 2013.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 67

Surreptitious Sharing on Android

Dominik Schürmann[1], Lars Wolf[1]

Abstract: Many email and messaging applications on Android utilize the Intent API for sharing images, videos, and documents. Android standardizes Intents for sending and Intent Filters for receiving content. Instead of sending entire files, such as videos, via this API, only URIs are exchanged pointing to the actual storage position. In this paper we evaluate applications regarding a security vulnerability allowing privilege escalation and data leakage, which is related to the handling of URIs using the *file* scheme. We analyze a vulnerability called *Surreptitious Sharing* and present two scenarios showing how it can be exploited in practice. Based on these scenarios, 4 email and 8 messaging applications have been analyzed in detail. We found that 8 out of 12 applications are vulnerable. Guidelines how to properly handle file access on Android and a fix for the discussed vulnerability are attached.

Keywords: Android, sharing, vulnerability, Intent, API.

1 Motivation

Android includes a rich API for communication and interaction between applications. It introduced the concept of Intents with a set of standardized actions to facilitate sharing of data. This allows applications to concentrate on its core functionality and rely on others to deal with extended use cases. For example, there is no need to support recording of video when implementing an email client, instead a video recorder can directly share a finished recording with the email application. In this example, it is crucial to only share the intended video—other recordings must be protected against unauthorized access. Android provides a variety of security mechanisms to implement access control. These include sandboxing on file system layer via Unix UIDs, as well as Android permissions and URI permissions on the API layer. Even though these mechanisms are already sophisticated in comparison to traditional desktop operating systems, several vulnerabilities have been discovered and discussed in the research community [EMM12, Mu14b, Mu15, Ba15a, Mi15]. These often emerge from edge cases in inter-application communication, which were not considered in the application developer's or Android's security model.

The main contribution of this paper is the presentation and evaluation of a security vulnerability we call *Surreptitious Sharing* related to content sharing via *file* schemes. So far this vulnerability has been neglected in literature and to the best of the authors' knowledge a related vulnerability exploiting *file* URIs has only been described in a security audit by Cure53 [He15]. We provide a detailed explanation of the vulnerability and analyze popular applications in regards to it. Finally, we will discuss best practices of securing Android against this issue and what developers can do to protect their users.

[1] TU Braunschweig, Institute for Operating Systems and Computer Networks, Mühlenpfordtstr. 23, 38106 Braunschweig, {schuermann, wolf}@ibr.cs.tu-bs.de

Since Android's inception and widespread adoption, a huge amount of research papers has been published analyzing certain security mechanisms. A great overview of Android's model and especially URI handling can be found in [EMM12]. An example of recent work in this area are publications of Bagheri et al. In [Ba15a], they developed a formal model to describe and evaluate Android's permission model to discover issues, e.g., privilege escalation attacks. They found a previously undiscovered flaw in Android's handling of custom permissions, which are still heavily based on installation order. They also found an URI permission flaw, where applications were able to access certain URIs because URI permissions are not properly revoked when the associated content provider has been uninstalled. In a different paper, they presented COVERT [Ba15b], a tool to evaluate the security of inter-application communication. Using formal models, the interactions of multiple applications can be checked based on the applications' control-flows. They showed their ability to find privilege escalation vulnerabilities using an experimental evaluation with applications from Google Play, F-Droid, MalGenome, and Bazaar. Still, it is not clear if the vulnerability presented in this paper could have been found using their model, which does not consider the underlying file system permissions.

While a vast amount of research has been published about static and dynamic analysis tools for Android, we will focus on specific discovered vulnerabilities related to the one presented in this paper. Mark Murphy discusses permission and URI permission issues in several blog posts. He describes a problem where an application can gain access to protected functionality by being installed before the targeted application, declaring the same custom permission as the targeted application while at the same time requesting the permission via `<uses-permission>` [Mu14b]. This vulnerability no longer works on Android 5; the installation of a second application requesting the same permission now fails with `INSTALL_FAILED_DUPLICATE_PERMISSION` [Mu14a]. In [Mu15], he discusses the limits of URI permissions granted via Content Providers.

A different way to gain access to data is by tricking the users to input their private information into user interfaces controlled by an attacker. Qi Alfred Chen et al. were able to launch malicious user interfaces from background processes to hijack login screens, i.e., launch a similar looking screen where the user enters her credentials [CQM14]. They were able to determine precise timings by detecting UI state changes via side channels of Android's shared memory [CQM14]. The authors of [NE13] tried to prevent other accidental data leakages on Android. They present Aquifer, a policy framework that developers can use to define restrictions and to protect their user interfaces in addition to Android's security model. They implemented an example using K-9 Mail, which has also been analyzed in this paper.

Other paper looking at application specific issues have been published. In [Fa13], 21 password manager have been evaluated. Besides complete failures in regards to cryptographic implementations, the main problem is that most password managers pasted passwords into Android's clipboard, which can be accessed without additional permissions. They propose fixes that require changes in the Android operating system, as well as fixes that can be deployed by developers. A more automated process of finding misused cryptographic principles has been investigated in [Eg13]. The authors found that 88% of all considered

applications violated at least one posed rule. An evaluation of the security of nine popular messaging tools has been presented in "Guess Who's texting You?" [Sc12]. It focuses on several aspects, such as deployed authentication techniques, registration mechanisms, and privacy leakages. A detailed analysis of WhatsApp can be found in [Th13].

A vulnerability similar to the one discussed in this paper has been discovered by Rob Miller of MWR Labs [Mi15]. He was able to bypass Android's file system permission model using Google's Android Admin application by opening a specially crafted webpage with this application. This webpage is opened inside the application's context and includes a *file* URI, which can be used to gain access to the private storage.

2 IPC and Access Control on Android

Before discussing the vulnerability in detail, we introduce fundamentals of Android's Inter Process Communication (IPC) and Access Control mechanisms. Android's IPC is based on the Binder mechanism. It is not based on traditional System V components, but has been implemented due to performance and security reasons as a patch for the Linux kernel. Memory can be allocated and shared between processes without making actual copies of the data. Binder's security is based on a combination of the usage of Unix UIDs and Binder capabilities. It is differentiated between direct capabilities, which are used to control access to a particular interface, and indirect capabilities, which are implemented by sharing tokens between applications. Android Permissions are checked by verifying that a permission that is associated to a Binder transaction is granted to the participating application. UIDs are used as an identifier in this process.

The Binder mechanism is normally not directly used inside the Android Open Source Project (AOSP) and Android applications. Instead, more easily usable IPC primitives exist that are based on Binder allowing communication between processes. These are Activity/Broadcast Intents, Messenger, and Services. While these primitives can be used to share data, they are only suitable for up to 1 MB of actual payload. Instead of sharing the data directly as a copy, it is advised to provide it via Content Providers and only share a Uniform Resource Identifier (URI) pointing to the payload. Google provides the class `FileProvider`[3] shipped with their support library to ease and secure the implementation of file sharing between applications. A file that is intended to be shared is temporarily stored inside the application's private cache directory and all metadata inside `FileProvider`'s database. A unique URI is created, and this URI is shared, for example via an Intent, to a different application. URIs served via `FileProvider` are using the *content* scheme, e.g., `content://com.google.android.apps.docs.files/exposed_content/6jn9cnzdJbDy`. Access to these files can be granted using two different methods. Calling `Context.grantUriPermission(package, Uri, mode_flags)` allows another package to retrieve the file until `revokeUriPermission()` is executed. Possible flags are `FLAG_GRANT_READ_URI_PERMISSION` and `FLAG_GRANT_WRITE_URI_PERMISSION`. If the URI is shared by Intent only, the flags can also be directly assigned via `Intent.setFlags()` which means that

[3] `http://developer.android.com/reference/android/support/v4/content/FileProvider.html`

access is granted to the called Activity until the corresponding stack is finished. Besides these *content*-URIs, Android supports a legacy way of sharing files, which is discouraged in recent documentations: URIs starting with the *file* scheme can point to an actual file in the file system, e.g., `file:///sdcard/paper.pdf`. The security of *file* URIs is based on file system permissions only, i.e., Unix UIDs. For the default case of accessing files on the external storage, access control is handled differently on Android versions: Since SDK 4 `WRITE_EXTERNAL_STORAGE` permission is required to access the card. In SDK 16, `READ_EXTERNAL_STORAGE` has been introduced for read operations only. Since SDK 19, actual external storages (not internal ones, which strangely also fall under the broader definition of "external storage") are no longer readable or writable. In SDK 21, the Storage Access Framework has been introduced which allows to access external storages via a new API, complementing the original Java `File` API. Since SDK 23, runtime permissions are introduced. `READ_EXTERNAL_STORAGE` permission must be granted during runtime to access *file* based URIs pointing to the SD card.

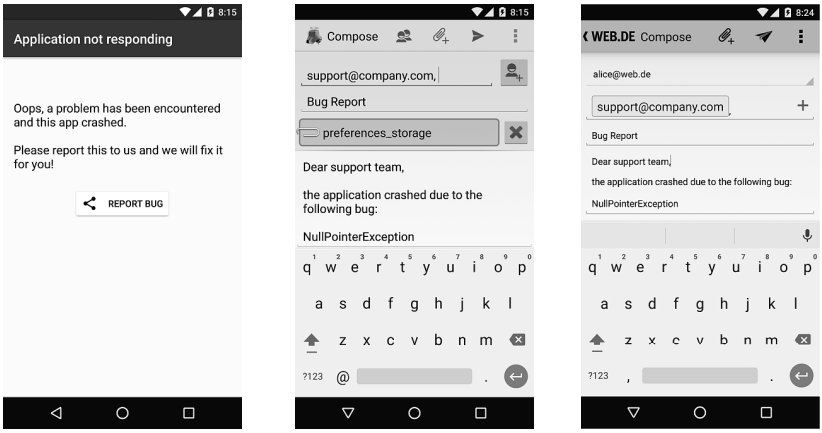
On the receiving side, these URIs are opened via `ContentResolver.openInputStream (Uri uri)`. This method supports the schemes *content*, *file*, and *android.resource* (used to open resources, e.g., `android.resource://package_name/id_number`).

In addition to storing on external mediums, applications can save private data using `Context.openFileOutput(String name, int mode)`. This datum is saved in `/data/data/com.example.app/` and can only be access by the application itself, not other applications or the user (in contrast to storing on SD cards)[4]. These methods utilize the standard File API and streaming classes of Java. Each application on Android has its own file system UID that protects these data directories against unauthorized access. In effect, all Android applications are separated from one another on the file system layer.

2.1 Sharing API

Using the described IPC methods, Android provides great ways to allow sharing of data between applications. One of the most simple standardized ways of sharing content between applications is by using share buttons or menus implemented in a variety of applications. By constructing an Intent with the `android.intent.action.SEND` action, a sender can transfer text or other content. The actual payload is included via so called extras, which are serialized using Android's Parcelable methods. For `SEND`, the following extras exist (all prefixed with `android.intent.extra.`): `TEXT`, `HTML_TEXT`, `STREAM`. While `TEXT` and `HTML_TEXT` contain the whole String, `STREAM` contains an URI, which points to the actual content via *content* or *file* schemes. Other standardized extras are `EMAIL`, `CC`, `BCC`, `SUBJECT`. Applications can introduce new extras, as Intent APIs are not fixed like traditional IPC interfaces. There exists a range of other Intent actions with similar semantics, such as `SEND_MULTIPLE`. It is basically the same mechanism as `SEND`, but allows sharing of multiple URIs using a list.

[4] `http://developer.android.com/guide/topics/data/data-storage.html#filesInternal`



(a) Pretended crash encouraging a bug report

(b) K-9 Mail

(c) WEB.DE

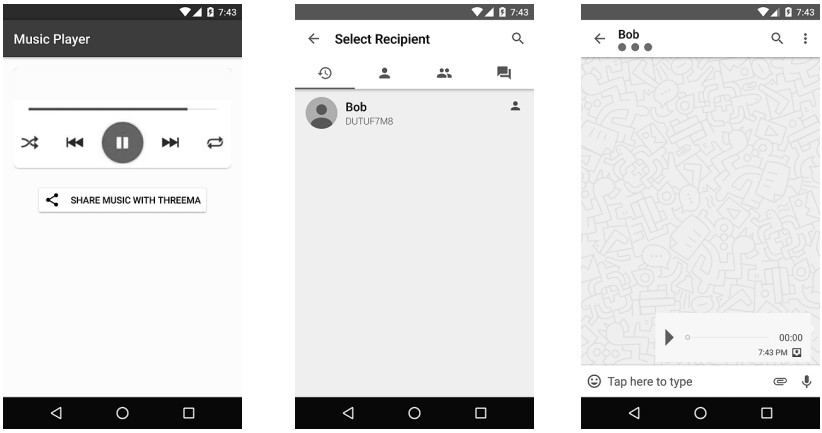
Fig. 1: Surreptitious Sharing of IMAP passwords stored in email clients (Scenario 1)

To receive data sent by these Intents, Android allows to specify Intent Filters inside the `AndroidManifest.xml`. Intent Filters consist of an XML structure defining the receiving action, the accepted MIME types, and filters which URIs are accepted[5]. It is important to note that explicitly declaring MIME types when constructing an Intent helps the receiving application to determine the type of content, but without any checks whatsoever that the actual payload is of that type. No *magic bytes* are checked by the Android OS.

3 Surreptitious Sharing

The central issue presented in this paper is related to the security of URIs using the *file* scheme. As discussed previously, access control to URIs based on this scheme is handled via traditional Unix file system permissions. The introduced permissions `READ_EXTERNAL_STORAGE` and `WRITE_EXTERNAL_STORAGE` are designed as additional barriers for accessing SD cards. The main issue lies in the fact that applications cannot only access their private data directories using `Context.openFileOutput(String name, int mode)`, but also using *file* URIs. While these URIs are normally used to access files on the SD card, via `file:///sdcard/paper.pdf` for example, they can also point to private files, e.g., `file:///data/data/com.example.app/files/paper.pdf`. If an application registers Intent Filters to support Android's sharing API or defines custom Intents accepting URIs, they are potentially accepting *file* URIs that could also point to their own private files. For applications facilitating communication, like email or messaging applications, this leads to what we call *Surreptitious Sharing*. To our knowledge, this issue has first been documented as vulnerability OKC-01-010 in Cure53's security audit of the OpenPGP application OpenKeychain [He15]. While their report applies this issue to the file

[5] `http://developer.android.com/guide/topics/manifest/intent-filter-element.html`



(a) Music player encouraging music sharing

(b) Threema: Select recipient

(c) Threema: Shared database shown as audio recording

Fig. 2: Surreptitious Sharing of messaging databases (Scenario 2)

encryption API in OpenKeychain, which accepts *file* URIs, we apply it in a broader context to communication applications. Investigating the AOSP source code reveals that support for *file* URIs using `Context.openFileOutput(String name, int mode)` (similar checks are present in `openAssetFileDescriptor`) was planned to be removed [6]. To demonstrate the impact of this security vulnerability, we consider two attack scenarios: Scenario 1) "Fake Bug Report" and Scenario 2) "Music Sharing".

Scenario 1 is intended to surreptitiously share IMAP passwords of email clients to an attacker. Following Figure 1, interaction with Scenario 1 consists of a sequence of actions. A malicious application shows a screen indicating that a problem has occurred urging the user to report the bug to the developers. Touching the button starts a malicious Intent specially crafted for a particular email application with an URI pointing to a private file of this email application, containing the IMAP password. Generally, this Intent uses the `SEND` action and a set of extras to prefill the recipient for example in addition to the malicious *file* URI inside the `STREAM` extra. The Intent is explicitly defined using the package name and the exact Activity. The private file is attached to the email shown in the compose screen. The user finally needs to send this email to expose her sensitive information.

Scenario 2 is designed to exploit messaging applications and share their databases to obtain message histories, for example. Instead of faking a crash followed by a bug report, it consists of a functional music player also featuring a button to share music with friends via installed messengers (cf. Figure 2). The button is prominently placed to encourage sharing. Based on the targeted messaging application, the user interfaces following the sharing action, differ.

[6] see inline comments in `openInputStream` method in `https://android.googlesource.com/platform/frameworks/base/+/master/core/java/android/content/ContentResolver.java`

Application	Version	Component	Open source	Displays filename	Ignores MIME type	File pre-processing	Security check	Secure
K-9 Mail	5.006	MessageCompose	●	●	●	○	○	○
AOSP Mail[a]	5.1.1-ecfa8c7973	ComposeActivityEmail	●	●	●	○	◐	○
GMail	5.8.107203005	ComposeActivityGmail	○	●	●	○	◐	○
WEB.DE	3.6	MailComposeActivity	○	●	●	○	○	○

[a] AOSP Mail from CyanogenMod 12.1

Tab. 1: Scenario 1 using `SEND` Intents with *file* URI

Even if it is not known beforehand which applications are installed, Android's API allows to query for available components via `PackageManager.queryIntentActivities()`. Thus, it is easy to search for targeted applications and then use a custom malicious Intent. In the following, we will investigate several applications in regards to this vulnerability. The applications have been evaluated on Android 6.0 (SDK 23), except AOSP Mail which has been tested on CyanogenMod 12.1 (Android 5.1, SDK 22). However, the vulnerability itself is present on all Android versions.

3.1 Scenario 1: Fake Bug Report

We will first look into two of the most popular open source email applications, K-9 Mail and AOSP Mail, and analyze their security. Due to the availability of their source code, it is possible to analyze the actual implementation of opening URIs more precisely. Afterwards, GMail and WEB.DE apps are tested. All results are listed in a concise form in Table 1.

K-9 Mail The crafted exploit (cf. Appendix 7.1) works as intended. As typical for an email client that accepts all MIME types (filters for "*/*"), no pre-processing is done for attached files and explicitly declared MIME types via `Intent.setType()` are ignored. The file `preference_storage`[7] containing the IMAP passwords is shared and displayed in K-9 Mail (cf. Figure 1b). At least, the attached file is displayed upfront to the user.

AOSP Mail Google has stopped introducing new features to the AOSP Mail client since the GMail application supports external IMAP accounts starting from version 5.0. Still, the AOSP Mail client is used as the code base for many derived proprietary clients and is still maintained by custom roms such as CyanogenMod. Here, we tested a version from CyanogenMod 12.1.
AOSP Mail has a security check in place to prevent attachments from data directories

[7] `file:///data/data/com.fsck.k9/databases/preferences_storage`

except from directories of the calling application, which shows "Permission denied for the attachment."[8]. By creating a world-readable hard link from the malicious application to the AOSP Mail's `EmailProvider.db`[9] file we were able to work around this security check and successfully attach the database containing the IMAP password (cf. Appendix 7.1). This also allows to hide the actual filename as only the name of the hard link is shown in the displayed attachments.

GMail GMail also has a similar security check in place resulting in "Permission denied for the attachment". Again, we were able to circumvent this by using a hard link. However, we were not able to exploit GMail on Android 6, maybe due to the new runtime permissions; this has not been investigated further. Retrieving the Google password is not possible due to the usage of OAuth in combination with the AccountManager API. Still, stored emails can be retrieved by retrieving the Google account name via `AccountManager.getAccounts()` and then sharing a URI pointing to `mailstore.example@gmail.com.db`[10].

WEB.DE The WEB.DE Android client is based on K-9 Mail and behaves similarly, except that the attachment is initially hidden behind the keyboard making it more difficult for the user to notice it (cf. Figure 1c).

3.2 Scenario 2: Music Sharing

Due to the nature of messaging applications, these are rarely used for bug reports. Thus, to exploit these, Scenario 2 is used. Table 2 gives an overview of the analyzed applications and their properties. We looked at five popular messaging applications from Google Play and three selected privacy-focused ones. In contrast to email applications, most messaging applications do not ignore the given MIME type and often rely on it instead of detecting the type itself.

WhatsApp We were not able to exploit WhatsApp. According to our analysis, only image, audio, and video types are supported. Sharing the message database[11] as an image MIME type resulted in "The file you picked was not an image file.", which seems to be detected during pre-processing of the image. Audio and video files were displayed in the message history, but were not properly uploaded; the retry button did not work.

Hangouts Hangouts behaves similarly to WhatsApp. Sending the database[12] as an image results in "Media format not supported".

[8] `https://android.googlesource.com/platform/packages/apps/UnifiedEmail/+/adea2c809c1a97d3948b131a6f36ee9b88039a45`
`https://android.googlesource.com/platform/packages/apps/UnifiedEmail/+/24ed2941ab132e4156bd38f0ab734c81dae8fc2e`

[9] `/data/data/com.android.email/databases/EmailProvider.db`

[10] `file:///data/data/com.google.android.gm/databases/mailstore.example@gmail.com.db`

[11] `file:///data/data/com.whatsapp/databases/msgstore.db`

[12] `file:///data/data/com.google.android.talk/databases/message_store.db`

Application	Version	Component	Open source	Displays filename	Ignores MIME type	File pre-processing	Security check	Secure
WhatsApp	2.12.365	ContactPicker	○	○	○[a]	●	○	●
Hangouts	6.0.107278502	ShareIntentActivity	○	○	○[a]	●	○	●
Facebook Messenger	50.0.0.11.67	ShareIntentHandler	○	◐	●[a]	◐	●	●[a]
Skype	6.12.0.585	SplashActivity	○	◐	●	◐	○	○
Snapchat	9.20.2.0	LandingPageActivity	○	○	●	●	○	●[a]
Threema	2.532	RecipientListActivity	○	◐	○	◐	○	○
Signal	3.5.2	ShareActivity	●	○	○	◐	○	○
Telegram	3.2.6	LaunchActivity	●	◐	○	◐	○	○

[a] Indefinite results, difficult to verify because application is not open source

Tab. 2: Scenario 2 using `SEND`, `SEND_MULTIPLE` Intents with *file* URIs

Facebook Messenger Facebook messenger registers the Intent for audio, video, image, text/plain and we were able to share a test file from the SD card, which was properly been sent and displayed in message history with its filename. However, we were not able to exploit the application. Sharing a malicious URI[13] from the internal storage resulted in "Sorry, Messenger was unable to process the file".

Skype Skype allows sharing of any MIME type and we were easily able to execute the exploit to retrieve `offlinestorage.db`[14]. Pre-processing is done for images and videos; sharing the malicious URI with these MIME types results in a frozen program.

Snapchat We were not able to retrieve private files from Snapchat, because it is limited to image files which are pre-processed before sending.

Threema As depicted in Figure 2, we were able to retrieve Threema's database[15] shown as an audio recording by setting the MIME type to "audio/mp4". This is possible because Threema does not pre-process these files before sending. Obviously, the audio cannot be played. The file can still be saved successfully by the receiving attacker and opened as a database file. Threema supports encrypted databases, which, according to [DLP13], are encrypted using a key saved beside the database. Using `SEND_MULTIPLE`, we were able to retrieve both the database and the key[16]. However, explicit MIME types are ignored here, thus the files cannot be hidden as audio recordings.

[13] `file:///data/data/com.facebook.orca/databases/threads_db2`
[14] `file:///data/data/com.skype.raider/files/offlinestorage.db`
[15] `file:///data/data/ch.threema.app/databases/threema.db`
[16] `file:///data/data/ch.threema.app/files/key.dat`

Signal Signal was in the same way vulnerable as Threema, i.e., its database[17] has been shared as an audio recording. Sending it using "image/png" resulted in a crash due to pre-processing in `AttachmentDatabase.getThumbnailStream()`. Because the fake image has been cached in Signal's database, Signal now crashes on each start.

Telegram Telegram was exploitable[18] but we were not able to hide the filename or type with any of the discussed tricks. However, using the hard link can help to make the user less suspicious.

4 Countermeasures

Upcoming Android versions should incorporate a security check into `ContentResolver.openInputStream()` that prevents opening of *file* URIs where the files are solely owned by the application's UID (cf. Appendix 7.2). In addition, Google's support library should be extended. While it contains a `FileProvider`[19] to provide files to other applications based on *content* URIs avoiding the *file* scheme, it should additionally include the method from Appendix 7.2 to securely open streams from URIs.

Furthermore, we recommend that application developers follow best practices to prevent Surreptitious Sharing: Content shared from other applications should be considered as unverified input that needs to be explicitly acknowledged by the user. Even after fixing the vulnerability, applications could still share different content than what has been shown to the user before, e.g., files from an SD card instead of the displayed image. Thus, we propose to pre-process content if possible as well as display the content and filename before actually sending it.

5 Conclusion

In this paper, we analyzed 12 applications in respect to a security vulnerability called Surreptitious Sharing. Our evaluation showed that 8 applications were exploitable and security checks implemented in GMail and AOSP Mail could be bypassed. Unfortunately, especially the privacy-focused messaging applications were easily exploitable. Hiding the private files by setting an explicit MIME type has been shown to work in Signal and Threema. Besides fixing the vulnerability, we recommend best practices for application developers how to handle shared files.

6 Acknowledgments

We thank Cure53 for discovering the original vulnerability OKC-01-010 [He15] and discussing possible attack vectors via hard links. We thank all vendors in advance for fixing the vulnerabilities.

[17] `file:///data/data/org.thoughtcrime.securesms/databases/messages.db`
[18] `file:///data/data/org.telegram.messenger/files/cache4.db`
[19] `http://developer.android.com/reference/android/support/v4/content/FileProvider.html`

References

[Ba15a] Bagheri, Hamid; Kang, Eunsuk; Malek, Sam; Jackson, Daniel: Detection of Design Flaws in the Android Permission Protocol Through Bounded Verification. In (Bjørner, Nikolaj; de Boer, Frank, Hrsg.): FM 2015: Formal Methods, Jgg. 9109 in Lecture Notes in Computer Science, S. 73–89. Springer, 2015.

[Ba15b] Bagheri, Hamid; Sadeghi, Alireza; Garcia, Joshua; Malek, Sam: COVERT: Compositional Analysis of Android Inter-App Permission Leakage. IEEE Transactions on Software Engineering, 41(9):866–886, Sept 2015.

[CQM14] Chen, Qi Alfred; Qian, Zhiyun; Mao, Z Morley: Peeking into your app without actually seeing it: Ui state inference and novel android attacks. In: Proceedings of the 23rd USENIX Security Symposium. S. 1037–1052, 2014.

[DLP13] Dimitrov, Hristo; Laan, Jan; Pineda, Guido: Threema security assessment. In: Research project for Security of Systems and Networks. Dezember 2013.

[Eg13] Egele, Manuel; Brumley, David; Fratantonio, Yanick; Kruegel, Christopher: An Empirical Study of Cryptographic Misuse in Android Applications. In: Proceedings of the 2013 Conference on Computer and Communications Security (CCS). ACM, New York, NY, USA, S. 73–84, 2013.

[EMM12] Egners, Andre; Meyer, Ulrike; Marschollek, Björn: Messing with Android's Permission Model. In: 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). S. 505–514, June 2012.

[Fa13] Fahl, Sascha; Harbach, Marian; Oltrogge, Marten; Muders, Thomas; Smith, Matthew: Hey, You, Get Off of My Clipboard. In: Financial Cryptography and Data Security, Jgg. 7859 in Lecture Notes in Computer Science, S. 144–161. Springer, 2013.

[He15] Heiderich, Mario; Horn, Jann; Aranguren, Abraham; Magazinius, Jonas; Weißer, Dario: Pentest-Report OpenKeychain, August 2015. `https://cure53.de/pentest-report_openkeychain.pdf`.

[Mi15] Miller, Rob: Sandbox bypass through Google Admin WebView. MWR Labs, August 2015.

[Mu14a] Murphy, Mark: Custom Permission Vulnerability and the 'L' Developer Preview. Commonsware Blog, August 2014. `https://commonsware.com/blog/2014/08/04/custom-permission-vulnerability-l-developer-preview.html`.

[Mu14b] Murphy, Mark: Vulnerabilities with Custom Permissions. Commonsware Blog, Februar 2014. `https://commonsware.com/blog/2014/02/12/vulnerabilities-custom-permissions.html`.

[Mu15] Murphy, Mark: The Limits of ContentProvider Security. Commonsware Blog, Juli 2015. `https://commonsware.com/blog/2015/07/13/limits-contentprovider-security.html`.

[NE13] Nadkarni, Adwait; Enck, William: Preventing accidental data disclosure in modern operating systems. In: Proceedings of the 2013 Conference on Computer and Communications Security (CCS). ACM, S. 1029–1042, 2013.

[Sc12] Schrittwieser, Sebastian; Frühwirt, Peter; Kieseberg, Peter; Leithner, Manuel; Mulazzani, Martin; Huber, Markus; Weippl, Edgar R: Guess Who's Texting You? Evaluating the Security of Smartphone Messaging Applications. In: 19th Annual Network & Distributed System Security Symposium (NDSS). 2012.

[Th13] Thakur, Neha S.: Forensic analysis of WhatsApp on Android smartphones, 2013.

7 Appendix

7.1 Example Exploit Targeting K-9 Mail

```
public void exploit() {
    Intent intent = new Intent();
    intent.setComponent(new ComponentName("com.fsck.k9", "com.fsck.k9.activity.
        MessageCompose"));
    intent.setAction(Intent.ACTION_SEND);
    String linkPath = createLink("/data/data/com.fsck.k9/databases/
        preferences_storage");
    Uri uri = Uri.parse("file://" + linkPath);
    intent.putExtra(Intent.EXTRA_STREAM, uri);
    intent.putExtra(Intent.EXTRA_EMAIL, new String[]{"support@company.com"});
    intent.putExtra(Intent.EXTRA_TEXT, "Dear support team,\n\nthe application
        crashed due to the following bug:\n\nNullPointerException");
    intent.putExtra(Intent.EXTRA_SUBJECT, "Bug Report");
    startActivity(intent);
}

private String createLink(String path) {
    File link = new File(getFilesDir().getPath() + "/bug_report");
    link.delete();
    try {
        Os.link(path, link.getAbsolutePath());
    } catch (ErrnoException e) {
        Log.e("SurreptitiousSharing", "hard link failed", e);
    }
    link.setReadable(true, false);
    return link.getAbsolutePath();
}
```

7.2 Secure Replacement for ContentResolver.openInputStream()

```
@TargetApi(VERSION_CODES.LOLLIPOP)
static InputStream openInputStreamSafe(ContentResolver resolver, Uri uri)
        throws FileNotFoundException {
    String scheme = uri.getScheme();
    if (ContentResolver.SCHEME_FILE.equals(scheme)) {
        ParcelFileDescriptor pfd = ParcelFileDescriptor.open(
                new File(uri.getPath()), ParcelFileDescriptor.parseMode("r"));
        try {
            final StructStat st = Os.fstat(pfd.getFileDescriptor());
            if (st.st_uid == android.os.Process.myUid()) {
                Log.e("SafeOpen", "File is owned by the application itself,
                    aborting!");
                throw new FileNotFoundException("Unable to create stream");
            }
        } catch (ErrnoException e) {
            Log.e("SafeOpen", "fstat() failed", e);
            throw new FileNotFoundException("fstat() failed");
        }

        AssetFileDescriptor fd = new AssetFileDescriptor(pfd, 0, -1);
        try {
            return fd.createInputStream();
        } catch (IOException e) {
            throw new FileNotFoundException("Unable to create stream");
        }
    } else {
        return resolver.openInputStream(uri);
    }
}
```

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 79

Comparative Evaluation of Machine Learning-based Malware Detection on Android

Sebastian Hahn[1], Mykolai Protsenko[2], Tilo Müller[3]

Abstract: The Android platform is known as the market leader for mobile devices, but it also has gained much attention among malware authors in recent years. The widespread of malware, a consequence of its popularity and the design features of the Android ecosystem, constitutes a major security threat currently targeted by the research community. Among all counter methods proposed in previous publications, many rely on machine learning algorithms based on statically extracted attributes from an app. Machine learning, which is also inspired by the developed field of desktop malware detection, has proven to be a promising approach for fighting Android malware. Many publications, however, rely on different data sets for different application attributes, rendering the comparison of them difficult. Furthermore, there exist attribute sets known from the desktop world which have not been ported to Android yet. In this paper, we aim to step towards filling this gap by assessing the effectiveness of the total number of 11 attribute sets, including those never evaluated on Android before, using a consistent data set of 10,000 apps. Our comparative evaluation provides a ranking for the single attribute sets according the detection performance they can reach, and suggests the most effective combination of all attributes.

Keywords: Android, Malware Detection, Machine Learning

1 Introduction

Due to the widespread of Android malware, a search for reliable detection methods remains an important research topic. In general, most detection measures can be classified as either being static or dynamic approaches. Static detection is usually facilitated by machine learning algorithms based on some static attributes of an application. In the numerous publications that emerged in the last years, researchers have proposed and evaluated many attribute sets for malware detection. At the same time, many static features are known to be valuable for machine learning based detection from the x86 domain, but have never been implemented for Android. In this paper we review those approaches and assess their effectiveness by performing an evaluation on a common dataset, which can bring us closer to the selection of the most optimal combination of static attributes for Android malware detection.

This paper is organized as follows: Section 2 gives some basic information about the Android system. Section 3 provides an overview of the attribute sets participated in our evaluation and covers details of their extraction implementation. The outcome of the

[1] Friedrich-Alexander University Erlangen Nuremberg, sebastian.hahn@informatik.stud.uni-erlangen.de
[2] Friedrich-Alexander University Erlangen Nuremberg, mykola.protsekno@fau.de
[3] Friedrich-Alexander University Erlangen Nuremberg, tilo.mueller@cs.fau.de

evaluation is given in Section 4 and the results are discussed in Section 5. Finally, we conclude in Section 6.

2 Background

This section covers the basics of the Android system. Readers familiar with Android can safely skip this section.

The Android mobile platform was created by the Open Headset Alliance with the goal to cope with the resource scarcity of hand-held devices and hardware plurality provided by numerous vendors. In a nutshell, Android is based on the Linux kernel enhanced by the core libraries and runtime environment, ontop of which the applications are executed. In the early versions of Android the runtime was represented by the Dalvik VM, a JIT supporting virtual machine with register-based bytecode which can be compiled from Java bytecode. Since Android version 5.0, Dalvik was replaced with ART, the more performant ahead of time compilation approach, generating machine code for each app from its bytecode at installation time. With both Dalvik and ART, the apps can also use native code by means of the Java native interface (JNI).

Within the code of any Android app, one can highlight the four main base classes which define the application components: Activities, Services, Broadcast Receivers and Content Providers. Each Activity corresponds to a single action performed by the app with a corresponding UI screen. The Services are used to perform some long-term background tasks and do not have any interface directly assigned to them. The Content Providers facilitate delivery of data by the app in response to the requests by other apps. The Broadcast Receivers help the application to define the reaction to the global events triggered either by the Android system or by installed apps. Examples for such events are a receipt of an SMS message or a low battery notification. The broadcast of such events, as well es invocation of Activities, relies on Intent objects, which can be considered as a description of a desired action.

Despite the recent replacement of the runtime environment, the structure of the app remained unchanged. The apps are shipped as zip-archived apk files, containing metadata, GUI layout definitions, Dalvik and native code, and other resources required by the app. The crucial metadata of the app is provided within the Android Manifest file and includes the description of the application components, the default Activity to be launched at the app startup, and the permissions requested by the apps.

The permissions play a crucial role in the Android security system. Each permission corresponds to a certain security-critical action, for instance dispatch of an SMS message. Each app declares the permissions it requires in its Manifest, and they are accredited by the user before app installation. Once being granted, an app may utilize its permissions at any time after.

3 Attributes and Implementation

All previously proposed solutions to the malware detection problem can roughly be classified in static, dynamic and hybrid approaches. The static ones analyze an app without its execution, the dynamic ones, on the opposite, aim to classify the app by monitoring its behavior at run time, and the hybrid approaches combine static and dynamic detection methods. In this paper, we focus on the static detection based on machine learning. A review of static attributes utilized in previously published classification approaches is provided in the following. An overview of the well-known dynamic and hybrid detection tools was performed by Neuner et al. [Ne14].

3.1 Manifest attributes

This subcategory of attributes is extracted from the Manifest which every apk has to contain. The Manifest is located in the main folder of the apk and contains various information regarding the application.

- **Permissions**: As mentioned in Section 2, Android permissions reflect the security-sensitive actions an app intends to make use of. Due to the fact that such actions are only accessible if the corresponding permissions have been granted, and the mandatory nature of their declaration, intuitively they can be very helpful in recognizing potentially malicious applications. Indeed, in many detection approaches known from the literature permissions play a crucial role [Sa13b, AZ13, Sa13a].
- **Features**: Features declare software and hardware properties on which an application depends. Unlike the permissions, the declaration of features is not necessary for an application to function properly and serves solely informational purposes, allowing to identify compatibility of an app with certain Android devices and system versions. The use of this attribute set was proposed by Santos et al. [Sa13a].
- **Intents**: This attribute is defined by the intents an app is using. Intents are used to either start an Activity, a Service, or to broadcast an event to all subscribed receivers. The use of Intents as attributes for malware detection was proposed by Arp et al. [Ar14]. An app components can only react to certain Intents if the corresponding `intent-filters` are declared in the Manifest. The filter declaration contains an action element, which describes what has to be performed in response to the received Intent or what global event has occurred. In this paper, we use the declared action as a representation of the Intent filters.
- **Application Components**: This feature set, utilized among other attributes by Arp et al. [Ar14], is defined by the application components, namely Activities, Services, and Broadcast Receivers, declared in the Android Manifest file. The Content Providers were not included in this set.

3.2 Code attributes

This set of attributes is extracted from the code contained in an app. If not stated otherwise, the following attributes refer to the bytecode of an app. The features derived from the native code are marked as such.

- **Used URLs**: This attribute composed from the URLs that are found inside the code of an app. The presence of certain URLs may help to identify malware since they can reveal command and control servers of botnets or resources to update and enhance malicious functionality. Since not every URL leads to a malicious website, we follow the example by Apvrille and Strazzere [AS12] and remove extremely frequently used URLs like the Android Market's URL, Google services, XML schemes and advertisement companies. Furthermore, the presence of URLs is taken into consideration regardless of the number of their occurrences.
- **Codesize**: The codesize feature is defined straight forward, it is the sum of the sizes of all codefiles in an apk. This feature was a part of the attribute set proposed by Apvrille and Strazzere [AS12].
- **Times of appearance of opcodes**:
  This attribute is based on the statistical properties of an app's code that was proposed by Santos et al. [Sa13a] to detect unknown x86 malware (PE). According to the authors, opcodes reveal significant statistical differences between malware and legitimate software. Following their example, for this attribute we compute the number of appearances of each opcode in the app's code.
- **Opcode sequences**: Similarly to the previous one, this attribute set is based on the occurrences of the bytecode instructions in an app's code, and was proposed for detection of Windows malware. Here, we apply the similar approach on the Android bytecode. In particular, we define attributes corresponding to the number of occurrences of the bytecode sequences of length two. Note that here only the instruction opcodes are considered, whereas the arguments of each instruction are discarded. Examples of the usage of such attribute for Android malware detection can be found in the works by Jerome et al. [Je14], Canfora et al. [CMV15], or Kang et al. [Ka13].
- **Presence of the native code**: Since the native code might be harder to analyze and is not supported by many analysis tools to the same extent as Dalvik bytecode, malicious apps may try to hide parts of their functionality in the native code. Furthermore, native code may be used to exploit vulnerabilities of the Android system, for instance to gain the root access to the device. Therefore, we add this attribute which reflects the presence of native libraries in the app. The use of the native code was also utilized as a feature for malware detection by Apvrille and Strazzere [AS12].

3.3 Other known attributes

This section provides an overview of the other attributes known form the literature, which do not fit in any of the categories above.

- **Presence of executables and zip-files in assets**: The use of this attribute set was suggested by Apvrille and Strazzere [AS12], motivated by the rational assumption that an attempt to hide data, e.g. an executable exploit, in the assets is a clear evidence for the app's malignity.
- **Statistical analysis attributes**: This statistical feature set was proposed by Tabish et al. [TSF09] for detection of x86 malicious files, not limited to executables only. In this paper we adapt this approach for Android malware detection, extracting the raw data from the files of the apk.
  Following the methodology described in the original paper, the content of each file is decomposed into blocks, which are used to form n-grams with various values of n. According to the authors [TSF09], the block size "plays a critical role in defining the accuracy", in this work it was set to 2000. The approach was evaluated for the same values of n as used by by Tabish et al., namely one, two, three and four.
  The feature set contains 13 items, which are computed exactly as described in the original paper, although other sources provide slightly different definitions of some values. The computation is performed for n-grams with n equals one, two, three, and four, resulting in a total number of 52 integer attributes in this set.

3.4 New attributes

Next we present the new features we propose in this paper, which to our best knowledge have never been used as attributes for Android malware detection.

- **Entropy of files in the resource-folder**: Previously we have introduced the attribute examining the presence of the executable or archived files in the assets of the app, originally proposed by Apvrille and Strazzere [AS12]. However, malicious payload can be hidden by more advanced means, e.g., using steganography. This attribute goes one step further in the search for suspicious contents of the apk. For this purpose we utilize entropy of the files in the resource folder as a measure of their 'structural order', with the expectation that unusual values will spot encrypted or otherwise alien content. To detect possible anomalies, we also calculate such statistical features of the entropy over all resource files as the arithmetic mean, the standard deviation and the empirical variance. On top of that we also take the maximum and minimum entropy values for each apk's resource file.
- **GUI Layout**: With this feature set we aim to recognize applications, similar by their visual appearance to the ones already known to the classifier. The attributes are extracted from the layout xml-files that define the structure of the app's GUI. These xml-files contain a hierarchy of layout items, for example LinearLayouts, Textboxes,

Buttons or Views. The features are defined as the total and the average number of each kind of Layout, Textbox, Button, and View in all xml-files of the apk.

- **Number of methods and number instructions per method**: This attribute should help the classifier to identify applications with similar code-structure as the known ones. For this purpose, from each apk we extract the following values: the number of methods in the code, the average number of opcodes in each method, and the standard deviation and empirical variance of the number of opcodes in each method. Note that the methods containing 3 or less opcodes are ignored, since we assume that those are getters or setters, or other short methods without relevant functionality.

3.5 Implementation

The extraction of the attributes described previously in this section was implemented ontop of the Androguard framework [DG11], which provided us with the necessary capabilities of processing the bytecode, as well as the Manifest and other xml files of the Android application package. For the classification of the samples we have utilized the WEKA tool [Ha09], which includes the classifiers that have demonstrated top performance in detecting Android Malware in previous studies: e.g., Random Forest, Bayesian Network, and K Nearest Neighbors.

4 Evaluation

This section is devoted to the practical evaluation of the attributes presented in Section 3. The evaluation contains the performance assessment of both single feature sets and combinations of the most promising attributes.

4.1 Dataset

The dataset we used for the evaluation was provided by Spreitzenbarth et al. and was collected within the MobileSandbox project [Sp13]. Out of the total number of obtained apps, we have randomly formed a dataset which contains 10,000 apps: 5,000 malicious samples and 5,000 benign ones.

For the purpose of the evaluation the dataset was randomly divided into a training- and a testset, in the proportion 80 to 20. Note that the testset does not contain any malware samples from the families included in the trainingset. This allows evaluation of the detection for previously unknown malware.

4.2 Performance Metrics

Next we describe the metrics of the detection quality which we utilized in our work to compare performance achieved by various classifiers with various attribute sets. The true

positive and true negative rates (TPR and TNR) correspond to the fraction of the correctly identified benign and malicious samples respectively. As an opposite, the values of false positives and false negatives (FPR and FNR) indicate the percentages of benign samples flagged as malicious and malicious samples assumed benign, respectively.

The true positive rate is sometimes called precision, meaning the ratio of the malware the classifier detected correctly amongst the total number of samples the classifier flagged malicious.

The accuracy can be considered 'the ratio of correct hits' and is defined as the sum of true positives and true negatives divided by the overall sample count in the test set. Sometimes instead of the false positive ratio its opposite is used, namely the specificity, as the ratio of true negatives to the total number of benign samples. Additionally, in our evaluation we include the time it took to build the classifier and the test time.

4.3 Single Feature Set Evaluation

This part of the evaluation aims to assess the detection performance of each single attribute set described previously in Section 3. The outcome is summarized in Table 1. According to the results, Android permissions are the best single predictor of the app's malignity, reaching the accuracy of about 96%. Very good classification quality was also achieved by opcode frequency, opcode sequences, and app components, with accuracy above 90%. For these and the most other attribute sets the best performance was shown by the Random Forest.

Such attributes as Intents, the presence of the native code or executable and archived files in the assets, as one could have expected, did not prove themselves as useful attributes on their own.

	Acc., %	TPR, %	FPR, %	Build Time, ms	Test Time, ms	Classifier
Permissions	96.02	96.00	3.95	6669	414	Random Forest
Opcode frequency	94.82	90.89	1.25	2196	321	Random Forest
Opcode sequences	94.82	90.29	0.65	10611	539	Random Forest
App components	94.27	91.25	2.7	199721	3628	Random Forest
Res. folder entropy	86.10	82.69	10.5	1606	95	Random Forest
Layout files	85.75	77.79	6.3	1359	233	Random Forest
Instr. per method	85.64	86.59	15.3	1337	108	Random Forest
Statistical analysis	79.92	75.44	15.6	2936	203	Random Forest
Codesize	76.12	79.34	27.1	3	394	KNN-5
Intents	74.24	50.82	3.26	3579792	3108	Random Forest
Features	52.63	98.75	92.92	258	169	Bayesian Network
Native code	51.74	08.70	5.25	8	7	Naive Bayes
Exec. and zip-files	50.26	98.40	97.85	8	11	Naive Bayes

Tab. 1: Ranking of the single attribute sets

4.4 Feature Combination Evaluation

As a second part of our evaluation, we have examined the performance of various combinations of the attribute sets. The results for the best of them are summarized in Table 2. The best performance with the accuracy over 97% was achieved by the combination of all attributes extracted from the Android Manifest, namely permissions, features, intents, and application components. Notably, we witnessed a slight accuracy increase if the features were excluded from this set.

In general, the detection performance based on permissions was improved if such attributes as the resource folder entropy, the opcode frequency or the layouts were added. On the other hand, the permissions performed better on their own than in combination with opcode sequences, application components, or statistical analysis attributes.

For the attribute set combinations the best classifier was also the Random Forest algorithm, showing the best performance in vast majority of cases including the top combinations.

	Acc., %	TPR, %	FPR, %	Build Time, ms	Test Time, ms	Classifier
Manifest attr. without feat.	97.35	96.95	2.25	108310	1793	Random Forest
Manifest attr.	97.30	96.65	2.05	105550	1755	Random Forest
Perm. and entropy	96.80	96.35	2.75	4744	477	Random Forest
Perm. and oc freq.	96.40	93.69	0.9	2600	348	Random Forest
Perm. and layout	96.30	95.15	2.55	3979	350	Random Forest
Perm., layout, oc freq.	96.30	93.54	0.95	2682	350	Random Forest
Perm. and oc seq.	95.57	96.45	5.3	172122	9	J48
Perm., app comp., oc freq.	95.35	91.64	0.95	17962	856	Random Forest
Perm. and stat. analysis	95.22	94.05	3.6	3186	393	Random Forest

Tab. 2: Ranking of the attribute set combinations by accuracy

5 Discussion

The results presented in the previous section indicate that the best performing attributes can be derived rather from an app's metadata stored in the Android Manifest than from the actual code of an app, although the code based attributes like opcode frequency and sequences also performed quite well. The reason is the nature of Android permissions which more or less precisely declare all the security-critical actions an app can possibly perform.

One the one hand, the Manifest attributes can provide a reliable and easy way to detect malicious apps, which does require neither static nor dynamic analysis of the code. On the other hand, the actual behavior of an app can be theoretically only derived from its code, a fact that inevitably leads to the possible drawbacks of the metadata-based detection: either a high false-positive rate, or high risk of missing truly malicious apps, i.e., low true positive

rate. Indeed, Table 1 shows, that the permissions on their own tend to have a comparatively high false positive rate.

The code attributes such as opcode frequencies and sequences, on the contrary, seem to be tightly bound with the actual behavior of an app defined by its code base, and therefore show much lower false positive rate as single attribute sets. Furthermore, in combination with permissions, opcode frequencies prove themselves as the most conservative attributes, with the least false positive rate.

The practical implementation of the malware detection tool based on our results has two options. First, it can be performed on device, which means an autonomous detection without Internet connection is possible, at cost of higher runtime overhead and the need of having the up-to-date trained classifier stored locally. Second, one could submit either the apk file itself, or the extracted attribute vector to a cloud for the analysis. Since in general it is associated with less overhead and it does not expose the classifier to the public, the second option seems to be more reasonable.

From the practical point of view, the main advantage of the strictly Manifest-based attributes would be the absence of the need to analyze code or other parts of the app, which would result in a much lower overhead. However, in our opinion, this advantage is outweighted by the disadvantage of having higher false positive rate, which is a quite important feature, as its high value would restrict users from installing benign applications.

Furthermore, for a practical anti-malware tool a test time would become a parameter of a higher importance. In our case, the best performing classifier Random Forest, despite having a high time consumption for training, can perform classification in relatively short amount of time.

6 Conclusion

In this paper we have examined different attributes according to their performance for machine learning based detection of Android Malware. In total, 11 previously known and new feature sets were evaluated as stand-alone attributes and in various combinations. Summarizing the results, we can point out that the leadership among the single feature sets belongs to Android permissions, which achieved accuracy of about 96%. For the combination of the attributes, the best performance was achieved by the following Manifest attributes: permissions, intents, and application components with over 97% accuracy. In both cases, the best accuracy was reached by the Random Forest classifier. Therefore, our results suggest the use of this combination either for a purely static detection approach, or as a 'reinforcement' of dynamic or hybrid tools like MobileSandbox [Sp13].

References

[Ar14] Arp, Daniel; Spreitzenbarth, Michael; Hübner, Malte; Gascon, Hugo; Rieck, Konrad; Siemens, CERT: Drebin: Effective and explainable detection of android malware in your pocket. In: Proceedings of the Annual Symposium on Network and Distributed System Security (NDSS). 2014.

[AS12] Apvrille, Axelle; Strazzere, Tim: Reducing the Window of Opportunity for Android Malware Gotta catch'em all. Journal in Computer Virology, 8(1-2):61–71, 2012.

[AZ13] Aung, Zarni; Zaw, Win: Permission-based Android malware detection. International Journal of Scientific and Technology Research, 2(3):228–234, 2013.

[CMV15] Canfora, Gerardo; Mercaldo, Francesco; Visaggio, Corrado Aaron: Mobile Malware Detection using Op-code Frequency Histograms. In: SECRYPT 2015 - Proceedings of the 12th International Conference on Security and Cryptography, Colmar, Alsace, France, 20-22 July, 2015. pp. 27–38, 2015.

[DG11] Desnos, Anthony; Gueguen, Geoffroy: Android: From Reversing to Decompilation. In: Black Hat Abu Dhabi. 2011.

[Ha09] Hall, Mark; Frank, Eibe; Holmes, Geoffrey; Pfahringer, Bernhard; Reutemann, Peter; Witten, Ian H.: The WEKA Data Mining Software: An Update. ACM SIGKDD Explorations Newsletter, 11(1):10–18, November 2009.

[Je14] Jerome, Q.; Allix, K.; State, R.; Engel, T.: Using opcode-sequences to detect malicious Android applications. In: Communications (ICC), 2014 IEEE International Conference on. pp. 914–919, June 2014.

[Ka13] Kang, Byeongho; Kang, BooJoong; Kim, Jungtae; Im, Eul Gyu: Android Malware Classification Method: Dalvik Bytecode Frequency Analysis. In: Proceedings of the 2013 Research in Adaptive and Convergent Systems. RACS '13, ACM, New York, NY, USA, pp. 349–350, 2013.

[Ne14] Neuner, Sebastian; der Veen, Victor Van; Lindorfer, Martina; Huber, Markus; Merzdovnik, Georg; Mulazzani, Martin; Weippl, Edgar R.: Enter Sandbox: Android Sandbox Comparison. In: Proceedings of the IEEE Mobile Security Technologies workshop (MoST). IEEE, IEEE, 05/2014 2014.

[Sa13a] Santos, Igor; Brezo, Felix; Ugarte-Pedrero, Xabier; Bringas, Pablo G: Opcode sequences as representation of executables for data-mining-based unknown malware detection. Information Sciences, 231:64–82, 2013.

[Sa13b] Sanz, Borja; Santos, Igor; Laorden, Carlos; Ugarte-Pedrero, Xabier; Bringas, Pablo Garcia; Álvarez, Gonzalo: Puma: Permission usage to detect malware in android. In: International Joint Conference CISISÂ12-ICEUTE´ 12-SOCO´ 12 Special Sessions. Springer, pp. 289–298, 2013.

[Sp13] Spreitzenbarth, Michael; Freiling, Felix; Echtler, Florian; Schreck, Thomas; Hoffmann, Johannes: Mobile-sandbox: Having a Deeper Look into Android Applications. In: Proceedings of the 28th Annual ACM Symposium on Applied Computing. SAC '13, ACM, New York, NY, USA, pp. 1808–1815, 2013.

[TSF09] Tabish, S Momina; Shafiq, M Zubair; Farooq, Muddassar: Malware detection using statistical analysis of byte-level file content. In: Proceedings of the ACM SIGKDD Workshop on CyberSecurity and Intelligence Informatics. ACM, pp. 23–31, 2009.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 89

SDN Malware: Problems of Current Protection Systems and Potential Countermeasures

Christian Röpke[1]

Abstract: Software-Defined Networking (SDN) is an emerging topic and securing its data and control plane is of great importance. The main goal of malicious SDN applications would be to compromise the SDN controller which is responsible for managing the SDN-based network. In this paper, we discuss two existent mechanisms aiming at protecting aforementioned planes: (i) sandboxing of SDN applications and (ii) checking for network invariants. We argue that both fail in case of sophisticated malicious SDN applications such as a SDN rootkit. To fill the corresponding security gaps, we propose two security improvements. The first one aims at protecting the control plane by isolating SDN applications by means of virtualization techniques. Compared to recent efforts, we thereby allow a more stringent separation of malicious SDN applications. The goal of the second proposal is to allow policy checking mechanisms to run independently from SDN controllers while minimizing hardware costs. Thereby, we improve SDN security while taking into account that correct functioning of policy checking can be manipulated by a compromised SDN controller.

Keywords: Software-defined networking, malicious SDN applications, SDN controller security.

1 Introduction

SDN is a new trend in the field of computer networking. It started in 2008 with OpenFlow [Mc08] which is a protocol for managing programmable switches by third-party control software. Since then, large companies such as HP, IBM and Brocade have introduced SDN-enabled switches [Opb] and researchers as well as enterprises have developed SDN control software (also known as *SDN controller* or *network operating system*) [Gu08, Fl, Po15, Sh14, Opc, ON, SDa, SDb, SDc]. Furthermore, multiple *SDN applications* are available, *e. g.*, in the industry's first SDN app store which is run by HP [HP13]. At the same time, a new research area has emerged including the use of SDN to enhance network security [Al15, Sh13a] as well as securing the SDN architecture [Po12, Sh13b, Sh14, RH15a, RH]. Nowadays, Gartner lists SDN as one of the top 10 strategic technology trends [Ga14a] and predicts that: "By the end of 2016, more than 10,000 enterprises worldwide will have deployed SDN in their networks" [Ga14b]. As SDN has increasingly attracted attention in both academia and industry, SDN has evolved from a research project to an emerging topic.

Technically, SDN decouples the control software, which decides where to forward network packets (*control plane*), from the forwarding hardware, which solely forwards network packets according to the made decision (*data plane*). As illustrated in Fig. 1, the

[1] Ruhr-University Bochum, Chair for System Security, Universitätsstrasse 150, 44801 Bochum, Germany, christian.roepke@rub.de

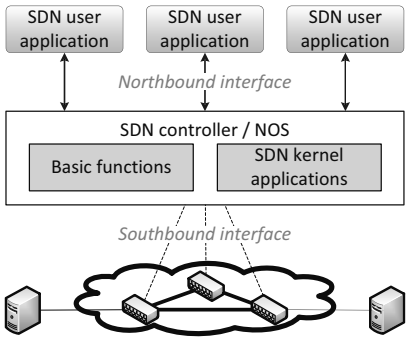


Fig. 1: SDN Architecture

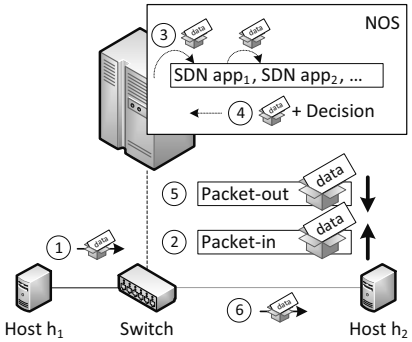


Fig. 2: OpenFlow Reactive Programming

decoupled control software is typically called *SDN controller* or *Network Operating System* (NOS). It is a logically centralized system with the main purpose of implementing network intelligence by programming the network. A SDN-based network thereby consists of programmable network devices and the interface between such devices and the NOS is typically called *southbound interface*. A popular southbound protocol is the aforementioned OpenFlow protocol which allows programming of network devices as well as requesting the network state. On top of a NOS, so called *SDN applications* implement network functionality as required by a network operator. In addition to the southbound interface, the interface between SDN controllers and SDN applications is called *northbound interface*. Inside a NOS, basic functionality is often extended by components typically also called as SDN applications. To distinguish between these two types of SDN applications, we call the ones using the northbound interface *SDN user applications* and the other ones *SDN kernel applications*. We thereby follow the naming convention of previous studies [Sh14, RH15b, RH].

The big benefit of SDN is that network operators can implement and adapt network functionality such as switching or routing independently from the network hardware vendor. For example, if a switch does not provide a feature which is needed or desirable by a network operator, such a feature can be implemented easily by using SDN to program network devices accordingly. In fact, industry is currently adopting SDN (17% of firms surveyed by Gartner already use SDN in production [Ga14b]) and large companies such as Google already benefit from OpenFlow to improve backbone performance as well as to reduce backbone complexity and costs [Hö12].

In SDN-based networks, a NOS plays a major role. It is responsible for programming network devices which can be achieved pro-actively as well as re-actively in case of OpenFlow. Proactive programming means that before a switch receives a specific type of network packets, the NOS (possibly triggered by a SDN application) adds one or more flow rules to this switch. Such flow rules tell the switch in advance how to forward matching network packets. In contrast, re-active programming works as illustrated in Fig. 2. This figure shows a typical SDN setup including an OpenFlow switch, which is controlled by a NOS and connects two hosts h_1 and h_2. In case host h_1 sends a packet towards host h_2

while the switch is missing a corresponding forwarding decision (*e. g.*, a flow rule telling the swich to forward the packet to host h_2), this packet (or only its headers) is delegated to the NOS via a so called *packet-in message*. In a next step, the NOS processes this packet typically by passing it to the set of SDN applications which have registered for such events. Depending on the NOS's implementation, each of these SDN applications inspects this packet-in message, for example, in a sequential order, aiming to determine an adequate forwarding decision. Finally, the NOS informs the switch of this decision via a so called *packet-out message* and, according to this decision, the switch forwards the packet to host h_2. Thus, this illustrated network programming highlights the central role of a NOS. From a security perspective, this essential SDN component is also an interesting place for SDN malware, for example, in the form of a malicious SDN application.

In this paper, we give an overview of existing SDN malware, describe already proposed countermeasures and reveal problems which can arise when using those. Furthermore, we propose improvements aiming at filling the resulting security gaps. In particular, we focus on two security mechanisms, namely, sandboxing and policy checking. Current sandbox systems restrict SDN applications to access only a certain set of critical operations while access to all other (non-critical) operations is not controlled. Thereby, SDN applications can be prohibited to perform operations which can harm the NOS, for example, by crashing the NOS or manipulating internal data structures. We also discuss policy checkers which can intercept network programming attempts in order to check if the consequent network state would violate a security policy. This mechanism allows to drop harmful programming attempts and, thereby, keeps the network policy-conform. Since both approaches have issues which can lead to security breaches, we present security mechanisms to improve protection of a NOS against malicious SDN applications. In particular, we discuss how SDN applications can be isolated from the NOS in a stronger way by means of virtualization techniques. We also propose a mechanism which allows for robust policy checking, meaning that it runs completely independent from a NOS while minimizing hardware costs.

2 Background

Before we go into details, we outline the existing SDN malware to give an insight about the current state of malicious SDN applications and their capabilities. Furthermore, we briefly describe the basic functioning of security mechanisms which are able to counteract the SDN applications with malicious logic (*i. e.*, sandboxing and policy checking).

2.1 Malicious SDN Applications

To the best of our knowledge, no SDN malware has been found in the wild yet. However, researchers have already demonstrated what kind of attacks can be performed by malicious SDN applications [Po12, Po15, Sh14, RH15a, RH, RH15b].

In [Po12, Po15], a new technique is presented allowing attackers to evade flow rules which are existent in an OpenFlow switch. By adding specially crafted malicious flow rules,

the authors demonstrate that an attacker can reach a network host although an existing drop rule explicitly denies such a connection. This technique is called *dynamic flow rule tunneling* and is based on standard OpenFlow instructions, *i. e.*, *set* and *goto*.

Other malicious SDN applications [Sh14, RH15a, RH] misuse critical operations on the system level in order to harm a NOS. For example, malicious SDN applications can easily crash SDN controllers, modify internal data structures or establish remote channels to retrieve shell commands from a C&C server, which in turn are executed on behalf of the NOS. In another example a malicious SDN application downloads and executes an arbitrary file with root privileges.

Recently, a SDN rootkit has been published [RH15b] for OpenDaylight which serves as a basis for multiple enterprise solutions [SDc]. This SDN rootkit heavily manipulates internal data structures and, thereby, gains control of the parts which are responsible for programming the network and reading the network's state. As a result, the authors demonstrate that an attacker is able to add and hide malicious flow rules as well as to remove legitimate flow rules without showing it to the administrator. Furthermore, an OpenFlow-based mechanism is presented which enables attackers to remotely communicate with the rootkit component running inside the NOS. This is interesting since a communication between hosts running on the data plane and the control plane is not a part of the SDN architecture.

2.2 Sandbox Systems

Currently, two sandbox systems for SDN controllers have been proposed. The first one runs SDN applications in separate processes and controls access to system calls [Sh14]. The other system utilizes Java security features to confine SDN applications inside of Java sandboxes [RH15a, RH, SDa]. The basic protection mechanism is illustrated in Fig. 3 and remains the same for both proposals.

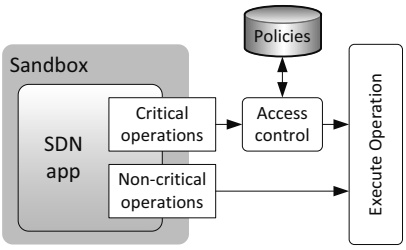


Fig. 3: Sandboxing

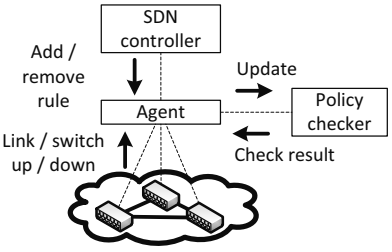


Fig. 4: Policy Checking

Each SDN application runs in a separate sandbox and access to critical operations (*i. e.*, system calls or sensitive Java operations) is controlled by a NOS. The sandbox configuration, *i. e.*, which SDN application is allowed to perform which critical operations, must be provided by a network administrator. In case he/she grants access to a critical operation, the corresponding SDN application can perform this operation while access is denied otherwise.

2.3 Policy Checking

Several studies introduce policy checking mechanisms for SDN-based networks [Ma11, KVM12, Kh12, Ka13]. The basic idea is to check whether the network state is conform to a given set of policies, for example, by looking for black holes, loops or security violations. While prior proposals struggle with performance issues, newer ones [Kh12, Ka13] provide the ability to perform policy checking at real time. This allows to intercept network programming attempts in order to check for policy violations first. Only in case of absent violations, corresponding flows rules are added to or removed from a network device. Fig. 4 illustrates this behavior for a policy checker called *NetPlumber* [Ka13] which is tested in Google's SDN-based WAN. In particular, a so called *NetPlumber agent* sits between the NOS and the programmable switches. This agent recognizes network state updates and in turn updates NetPlumber's internal model of the network. Such updates are triggered when a NOS adds or removes a flow rule and when a link or switch gets up or down. On each of these events, policy checks are initiated aiming at alerting an administrator in case a given policy is violated. Note that such an alert can occur before a policy violation takes place (*i. e.*, when adding or removing a flow rule) or soon after it (*i. e.*, when a link/switch gets up/down).

3 Problems of Current Protection Systems

3.1 Sandbox Systems

Concerning sandbox systems, malicious SDN applications can use potentially harmful operations only if access to them is granted explicitly. Current systems [Sh14, RH15a], however, support only low-level critical operations, *i. e.*, system calls and sensitive Java operations. This means that an administrator must have special knowledge in system security especially regarding operating system functioning and the Java programming language in order to estimate security-related consequences of granting access to a critical operation. Thus, the use of low-level based sandbox systems may lead to security breaches in practice as network administrators are rather experts in network than in system security and 50% to 80% of network outages are caused by misconfigurations [Ju08].

A recently proposed sandbox system [SDa] includes not only low-level but also high-level critical operations such as *reading the network topology* or *receiving a packet-in message*. Such operations are rather easy to understand by network administrators and corresponding security issues can be estimated more easily. However, supporting high-level critical operations does not mean that access to low-level critical operations can be denied by default. For example, SDN applications may depend on accessing the file system (*e. g.*, writing temporary files or reading configuration files), establishing network connections (*e. g.*, retrieving information stored on another network host) or using runtime capabilities such as creating processes or threads (*e. g.*, for concurrent programming or parallel execution). Denying access to such low-level critical operations by default would heavily limit SDN application developers in the way of implementing network functionality – which

is the main purpose of a SDN application. In the best case, a sandbox system can deny access to a pre-configured set of low-level critical operations while allowing the administrator to decide on the remaining critical operations separately. However, this also limits SDN application developers to a level which may not be feasible in practice.

Even if we assume that network administrators are experts in both network and system security, it is difficult to determine the way (*i. e.*, benign or malicious) a critical operation is used in. But access to a critical operation can only be denied by default if it is exclusively used in a malicious way. To illustrate the problem, let us have a look at Java reflection. It can be used to read or manipulate internal data structures of a Java-based NOS. Because of this, one might guess that access to corresponding critical operations should be denied by default. In fact, Java reflection has already been used maliciously to implement a SDN rootkit [RH15b]. However, legitimate SDN applications may also use Java reflection but in a benign way. In particular, we have analyzed several SDN kernel applications provided by the HP app store. At the time of writing, we find that 6 out of 11 examined SDN applications also use Java reflection. At a first glance, we could not find a single Java reflection related critical operation which is used by the aforementioned SDN rootkit but never by the benign SDN applications.

Consequently, configuration of current sandbox systems with malicious logic prevention is rather difficult. Taking this into account, network administrators may require more effective systems in order to improve protection of a NOS against malicious SDN applications.

3.2 Policy Checking

With regard to policy checking, adverse network manipulations (*e. g.*, a malicious flow rule) can be prohibited by checking at real time if a network event (*e. g.*, adding a new flow rule) violates one or more given security policies. Current real time policy checkers run either inside [Kh12] or outside a NOS [Ka13]. Since malicious SDN applications (such as a SDN rootkit) can compromise a NOS including its security checks [RH15b], we focus on policy checking performed outside the NOS.

For example, NetPlumber [Ka13] is implemented as a network service which is externally triggered by a NetPlumber agent to perform policy checks. Where this agent should be implemented is not discussed in detail by the authors. In fact, they simply use local files instead of network connections in order to load network updates into the NetPlumber system during evaluation. In general, there are several options where such an agent can be implemented: (i) inside a NOS, (ii) outside a NOS but running on the same host and (iii) outside a NOS and running on a separate host. In the first case only a few additional changes within a NOS are necessary for triggering policy checking before a flow rule is sent to a switch. Obviously, this can be achieved easily and cost-effective. The second option is more complicated but still does not require additional hardware as a separate proxy process or network packets interception on kernel level can achieve network programming interception. Most complex and expensive is the last option: running the agent on a separate hardware. For example, a backup component for a NOS is required to handle

breakdowns or maintenance-related shutdowns. Another cost-driving factor is the network interface connecting SDN switches and a NOS, which must handle network events at high speed [Ka09]. Furthermore, in case of long distance networks, SDN controllers must run at different locations to avoid latency issues [HSM12]. Additional hardware for such scenarios and extra complexity in terms of additional operational costs can cause additional expenses.

As a major goal of SDN is cost degression, such additional expenses may not be an option for network operators. However, running policy checks independently from the NOS is highly desirable for security reasons.

4 Potential Countermeasures

In order to fill the security gaps caused by the aforementioned problems, we propose the following countermeasures.

4.1 Strong Isolation of SDN Applications

Considering the issues described in Section 3.1, we need a mechanism which counteracts malicious SDN applications while assuming that sandboxing is insufficient in practice. In order to compensate that, we propose the following NOS architecture which is based on isolation of SDN applications by means of virtualization techniques. This is new with respect to SDN controllers and it enables a more stringent way of separating SDN applications compared to existing systems.

Our architecture is illustrated in Fig. 5 and includes several protection domains each running separately, for example, in a virtual machine. Inter-domain communication is only allowed between the SDN controller domain and each application domain while this communication is restricted to high-level and SDN-related operations. Assuming that virtualization presumes domain isolation, the big benefit of this architecture is that critical low-level operations are executed within each protection domain, thus, negative consequences are limited to the affected domain only.

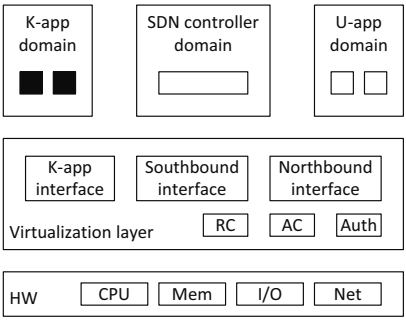


Fig. 5: Strong Isolation of SDN Apps

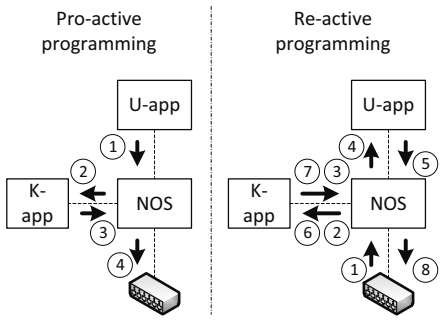


Fig. 6: Critical Data Path

In particular, we run one or more SDN applications in a separate application domain instead of starting them in separate processes. Thereby, SDN user applications (U-apps) communicate via the northbound interface with the SDN controller, SDN kernel applications (K-apps) use a NOS-specific interface (K-app interface) for communication with the NOS, and the SDN controller uses the southbound interface to interact with the SDN-based network. In addition, a virtualization layer provides Authentication (Auth), Resource Control (RC) and Access Control (AC) mechanisms allowing to authenticate components, assign hardware resources to each domain and control access to operations related to the aforementioned interfaces. In case of the malicious use of critical operations, this architecture limits negative consequences in a way not available in current systems. For example, if a SDN rootkit uses Java reflection and the corresponding sandbox allows such critical operations, today's Java-based SDN controllers can be compromised despite the use of sandboxes. Given our architecture, this is not possible while SDN application developers remain unrestricted in inventing new applications.

The main challenge in this respect is to implement network programming efficiently as this is the main purpose of a NOS. Therefore, Fig. 6 shows the critical data paths for pro-active and re-active network programming, separately. In case of pro-active programming, a flow rule generated by a SDN user application must be transferred from an application-domain to the controller-domain (①). Depending on the SDN controller, a SDN kernel application may be used to receive this flow rule (②), encapsulate it in a southbound protocol dependent format and send it back to the NOS (③). Then, the NOS must transmit this packet to the affected switch(es) (④). In case of re-active programming, the critical data path consists of some additional steps. When a network packet is delegated to a NOS (①), the NOS must pass it to the component which implements the southbound protocol, for example, a SDN kernel application (②). After extracting the needed information, it must be passed to the SDN user applications which are registered for such tasks (③ and ④). Next, after determining an adequate forwarding decision it must be sent to the affected switches which works similar to pro-active programming (⑤, ⑥, ⑦ and ⑧). As performance of network programming is of high importance, the biggest challenge is to reduce the overhead in order to achieve practicable results.

4.2 Robust Policy Checking

Checking the network state against policy violations at real time is challenging especially in case of a compromised SDN controller. Besides running policy checker agents on separate hardware as described in Section 3.2, we propose an alternative aiming at saving the hardware costs. As illustrated in Fig. 7(a) and 7(b), we introduce a policy checking mechanism which is completely independent from a NOS, thus, robust against a compromised NOS. The basic idea is to treat a new flow table entry as inactive until a policy checking instance signals that no policy is violated.

Fig. 7(a) illustrates the work flow of adding a flow rule. In this case, a SDN controller programs network devices as usual, for example, via OpenFlow (①). However, the corresponding programming command not only adds a flow table entry but also marks it as

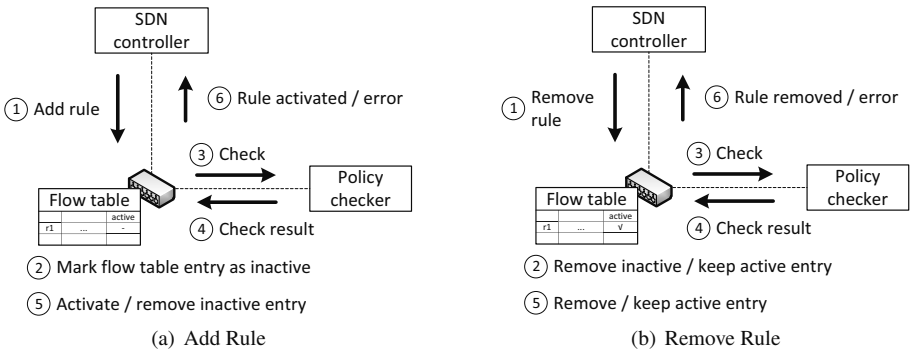


Fig. 7: Robust Policy Checking

inactive (②). In the next step, a switch acts as a policy checker agent and triggers a policy checker to validate the absence of policy violations when adding a new flow rule (③). Depending on the check result (④), the switch either activates the flow table entry or removes it from the flow table (⑤). In the final step, the switch responds either by an error message or by a *flow activated* message telling the NOS that the programmed flow rule is policy-conform (⑥). In case of removing a flow rule, the work flow remains similar as shown in Fig. 7(b). Receiving a command to remove a flow rule (①), the switch checks if the corresponding flow table entry is active or not (②). In case of inactive entry, the switch removes it and returns a *flow removed* message to the NOS (⑥). In the opposite case, the flow table entry remains active until policy checker signals no policy violations by removing the flow rule (③ and ④). After this confirmation, the active entry is removed (⑤) and a flow removed message is sent to the NOS (⑥). In case of a policy violation, the corresponding flow entry remains active and a corresponding error message is sent to the NOS (⑥).

In contrast to current real time policy checkers, the mechanism does not need extra hardware in order to run completely independent of a NOS. In case of OpenFlow, however, this mechanism requires several changes: (i) OpenFlow must support roles not only for SDN controllers (*e. g.*, master and slave) but also for policy checkers; (ii) flow table entries must be extended by a field indicating active vs. inactive entries; (iii) processing of network packets must be adapted such that only active flow table entries are considered when finding a matching flow table entry; and (iv) OpenFlow switches must interact with a policy checker in order to process network programming. For reliability reasons, a switch should be able to interact with multiple policy checkers to avoid single points of failures.

5 Related Work

The problem of malicious SDN applications was first addressed by Porras *et al.* [Po12]. The authors present a new technique called *dynamic flow rule tunneling* which enables attackers to bypass existent flow rules by exploiting standard OpenFlow instructions. Later on, Shin *et al.* [Sh14] and Röpke *et al.* [RH15a] present SDN applications with rudimentary

malicious logic. For this purpose, the authors exploit the fact that many SDN controllers allow a SDN application to execute any kind of critical operations without limiting access to them. Recently, Röpke *et al.* [RH15b] demonstrate how a sophisticated SDN malware can compromise SDN controllers. In particular, the authors provide a SDN rootkit for the popular OpenDaylight controller which serves as a foundation for several enterprise solutions.

Despite the fact that no malicious SDN application has been found in the wild yet, we take the findings of previous studies as examples of upcoming attack scenarios. This motivated us to examine current protection mechanisms regarding their efficiency against sophisticated malicious SDN applications. Sandboxing as well as policy checking have been already discussed by Röpke *et al.* [RH15b], but we discuss the potential problems in detail and present valuable solutions.

Our first proposal is mainly motivated by security systems such as Qubes [RW10] and Bromium [Br] which use virtualization techniques to isolate applications from common operating systems like Linux and Windows. Note that using such a technique does not provide isolation properties by default as it was shown recently [XE]. However, vulnerabilities which allow attackers to break out of a VM are rather rare while escaping from a process environment is commonly used by malware, for example, to infect other processes. Our second proposal is motivated by the fact that a compromised SDN controller can manipulate policy checking [RH15b]. We therefore extend OpenFlow [Opa] in order to trigger policy checks independently from a SDN controller. One may think that adding functionality to a network device is against the paradigm of SDN which aims at decoupling control functions from such elements. However, the OpenFlow specification already includes functions like checking for overlapping flow rules. Moreover, adding security functions on the data plane has been implemented efficiently before [Sh13b].

6 Conclusions

In this paper, we discuss problems of state-of-the-art security mechanisms which help in protecting SDN controllers against malicious SDN applications. We argue that current sandbox systems and policy checking mechanisms indicate problems when considering sophisticated SDN malware such as a SDN rootkit. In order to fill the corresponding security gaps, we propose two improvements. On the one hand, we introduce a SDN controller design based on virtualization techniques which enables a strong isolation of SDN applications from a SDN controller. On the other hand, we present an enhanced way of policy checking which runs independently from the SDN controller and remains robust even if a NOS gets compromised. Each of our proposals improves the security of SDN-based networks especially against malicious SDN applications. This is particularly important as we believe that SDN malware will appear soon considering the fast development of SDN.

References

[Al15] Ali, Syed Taha; Sivaraman, Vijay; Radford, Adam; Jha, Sanjay: A Survey of Securing Networks using Software Defined Networking. IEEE Transactions on Reliability, 2015.

[Br] Bromium: , Bromium. www.bromium.com.

[Fl] Floodlight project: , Floodlight. floodlight.openflowhub.org.

[Ga14a] Gartner: , Gartner Identifies the Top 10 Strategic Technology Trends for 2015. http://www.gartner.com/newsroom/id/2867917, 2014.

[Ga14b] Gartner: , Predicting SDN Adoption. http://blogs.gartner.com/andrew-lerner/2014/12/08/predicting-sdn-adoption/, 2014.

[Gu08] Gude, Natasha; Koponen, Teemu; Pettit, Justin; Pfaff, Ben; Casado, Martín; McKeown, Nick; Shenker, Scott: NOX: Towards an Operating System for Networks. ACM SIGCOMM Computer Communication Review, 2008.

[HP13] Hewlett-Packard: , HP Open Ecosystem Breaks Down Barriers to Software-Defined Networking. www8.hp.com/us/en/hp-news/press-release.html?id=1495044#.Vh-AbR_HnCI, 2013.

[HSM12] Heller, Brandon; Sherwood, Rob; McKeown, Nick: The controller Placement Problem. In: ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. 2012.

[Hö12] Hölzle, Urs: , OpenFlow @ Google. Open Networking Summit, 2012.

[Ju08] Juniper Networks: , What's behind network downtime? www-935.ibm.com/services/au/gts/pdf/200249.pdf, 2008.

[Ka09] Kandula, Srikanth; Sengupta, Sudipta; Greenberg, Albert; Patel, Parveen; Chaiken, Ronnie: The nature of data center traffic: measurements & analysis. In: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference. 2009.

[Ka13] Kazemian, Peyman; Chang, Michael; Zeng, Hongyi; Varghese, George; McKeown, Nick; Whyte, Scott: Real Time Network Policy Checking Using Header Space Analysis. In: USENIX Symposium on Networked Systems Design and Implementation. 2013.

[Kh12] Khurshid, Ahmed; Zhou, Wenxuan; Caesar, Matthew; Godfrey, P: Veriflow: Verifying Network-Wide Invariants in Real Time. ACM SIGCOMM Computer Communication Review, 2012.

[KVM12] Kazemian, Peyman; Varghese, George; McKeown, Nick: Header Space Analysis: Static Checking for Networks. In: USENIX Symposium on Networked Systems Design and Implementation. 2012.

[Ma11] Mai, Haohui; Khurshid, Ahmed; Agarwal, Rachit; Caesar, Matthew; Godfrey, P; King, Samuel Talmadge: Debugging the Data Plane with Anteater. ACM SIGCOMM Computer Communication Review, 2011.

[Mc08] McKeown, Nick; Anderson, Tom; Balakrishnan, Hari; Parulkar, Guru; Peterson, Larry; Rexford, Jennifer; Shenker, Scott; Turner, Jonathan: OpenFlow: Enabling Innovation in Campus Networks. ACM SIGCOMM Computer Communication Review, 2008.

[ON] ONOS Project: , Open Network Operating System. onosproject.org.

[Opa] Open Networking Foundation: , OpenFlow Switch Specification, Version 1.4.0. www.opennetworking.org.

[Opb] Open Networking Foundation: , SDN/OpenFlow Products. www.opennetworking.org/sdn-openflow-products.

[Opc] OpenDaylight Project: , OpenDaylight. www.opendaylight.org.

[Po12] Porras, Philip; Shin, Seungwon; Yegneswaran, Vinod; Fong, Martin; Tyson, Mabry; Gu, Guofei: A Security Enforcement Kernel for OpenFlow Networks. In: ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking. 2012.

[Po15] Porras, Phillip; Cheung, Steven; Fong, Martin; Skinner, Keith; Yegneswaran, Vinod: Securing the Software-Defined Network Control Layer. In: Symposium on Network and Distributed System Security. 2015.

[RH] Röpke, Christian; Holz, Thorsten: On Network Operating System Security. To appear in International Journal of Network Management.

[RH15a] Röpke, Christian; Holz, Thorsten: Retaining Control Over SDN Network Services. In: International Conference on Networked Systems (NetSys). 2015.

[RH15b] Röpke, Christian; Holz, Thorsten: SDN Rootkits: Subverting Network Operating Systems of Software-Defined Networks. In: Symposium on Recent Advances in Intrusion Detection. 2015.

[RW10] Rutkowska, Joanna; Wojtczuk, Rafal: Qubes OS architecture. Invisible Things Lab Tech Rep, 2010.

[SDa] SDN security project: , Secure-Mode ONOS. sdnsecurity.org.

[SDb] SDxCentral: , Comprehensive List of SDN Controller Vendors & SDN Controllers. www.sdxcentral.com/resources/sdn/sdn-controllers/sdn-controllers-comprehensive-list/.

[SDc] SDxCentral: , SDN Controllers Report. www.sdxcentral.com/reports/sdn-controllers-report-2015/.

[Sh13a] Shin, Seungwon; Porras, Phillip; Yegneswaran, Vinod; Fong, Martin; Gu, Guofei; Tyson, Mabry: FRESCO: Modular Composable Security Services for Software-Defined Networks. In: Symposium on Network and Distributed System Security. 2013.

[Sh13b] Shin, Seungwon; Yegneswaran, Vinod; Porras, Phillip; Gu, Guofei: AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software-Defined Networks. In: ACM Conference on Computer and Communications Security. 2013.

[Sh14] Shin, Seungwon; Song, Yongjoo; Lee, Taekyung; Lee, Sangho; Chung, Jaewoong; Porras, Phillip; Yegneswaran, Vinod; Noh, Jiseong; Kang, Brent Byunghoon: Rosemary: A Robust, Secure, and High-Performance Network Operating System. In: ACM Conference on Computer and Communications Security. 2014.

[XE] XEN project: , Security Advisory XSA-148. xenbits.xen.org/xsa/advisory-148.html.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 101

Auf dem Weg verTAN: Über die Sicherheit App-basierter TAN-Verfahren

Vincent Haupert[1], Tilo Müller[2]

Abstract: Die deutschen Banken wenden sich zunehmend von den alten TAN-Verfahren ab. Als Motiv zur Erschließung neuer Techniken abseits der indizierten TAN-Liste, mTAN und chipTAN wird neben der Sicherheit auch der fehlende Komfort durch die Notwendigkeit dedizierter Hardware angeführt. Neue App-basierte TAN-Verfahren erlauben es dem Nutzer, eine Transaktion mit seinem mobilen Endgerät (Android oder iOS) auszulösen und *auf dem selben Gerät* zu bestätigen – und das bei vermeintlich höherer Sicherheit als bei den etablierten Verfahren. Wir haben die Sicherheit solcher App-basierten TAN-Verfahren am Beispiel des pushTAN-Verfahrens der Sparkassen ausgewertet und attestieren dem Verfahren gravierende konzeptionelle Schwächen. Der bewusste Verzicht auf eigenständige Hardware zur Transaktionsauslösung und -bestätigung macht das Verfahren für Schadsoftware zu einer leichten Beute. Zur Demonstration dieser Schwächen haben wir einen Angriff entwickelt, der vom Nutzer Transaktionen abfängt und vor ihrer Bestätigung nach Belieben manipulieren kann.

Keywords: Zwei-Faktor-Authentifikation, Onlinebanking, TANs, Android Reverse Engineering

1 Einleitung

Onlinebanking wird heute von über zwei Drittel der deutschen Internetnutzer zur Tätigung ihrer Bankgeschäfte genutzt [e.V15]. Um den Onlinezugang eines Nutzers vor Missbrauch zu schützen, muss jede Transaktion durch eine Transaktionsnummer (TAN) bestätigt werden. Hierfür bieten die Kreditinstitute ihren Kunden meistens verschiedene Verfahren zur Auswahl an, die sich in ihrer Funktionsweise, ihrer Sicherheit und ihrem Komfort unterscheiden. Neue App-basierte TAN-Verfahren sollen die etablierten Verfahren ablösen und Onlinebanking bequemer und sicherer machen. Die Notwendigkeit von dedizierter Hardware entfällt und Transaktionen können mit einem einzelnen mobilen Endgerät durchgeführt werden. Sowohl die Sparkassen, Volksbanken-Raiffeisenbanken, die Hypovereinsbank, als auch die DKB und die ING-DiBa haben entsprechende Apps im Angebot. Es ist anzunehmen, dass die übrigen deutschen Kreditinstitute folgen werden, um ihren eigenen Kunden denselben Komfort bieten zu können und nicht zuletzt, weil vor Kurzem ebenfalls Angriffe auf die alternative mTAN bekannt geworden sind [Fre15].

Der Mobilitätsaspekt von Produkten zur App-basierten TAN-Erzeugung wird von den Banken ebenso betont, wie eine durch den TÜV abgelegte Sicherheitszertifizierung. Das pushTAN-Verfahren der Sparkassen sei speziell gehärtet, arbeite mit kryptographischen Verfahren und Signaturen und werde zudem isoliert in einem separaten Kanal betrieben.

[1] FAU Erlangen-Nürnberg, Department Informatik, Martensstr. 3, 91058 Erlangen, vincent.haupert@cs.fau.de
[2] FAU Erlangen-Nürnberg, Department Informatik, Martensstr. 3, 91058 Erlangen, tilo.mueller@cs.fau.de

Auf ihrer Internetpräsenz listet die Sparkasse „Mobile-Banking mit der pushTAN“ sogar als erstes Verfahren und erweckt bei ihren Lesern damit den Eindruck, als sei pushTAN besonders zu empfehlen, oder zumindest gleichwertig gegenüber den etablierten Verfahren. Ein erhöhtes Gefahrenpotential durch pushTAN wird weder von den Sparkassen aufgeführt, noch ist uns eine unabhängige kritische Analyse durch Dritte bekannt. Im Gegenteil, nachdem der TÜV es als sicher eingestuft hat, die Banking-App zur Transaktionsauslösung und die TAN-App zur Transaktionsbestätigung auf ein und dem selben Gerät zu verwenden, haben sich die Banken trotz aller Schutz- und Härtungsmaßnahmen de facto von der Zwei-Faktor-Authentifikation verabschiedet — ohne dass dies von Sicherheitskreisen angeprangert oder diskutiert worden wäre.

2 Die Zwei-Faktor-Authentifikation im Onlinebanking

Seit jeher ist Onlinebanking ein Zwei-Faktor-Verfahren. Schon als es 1980 die Verbraucherbank – damals noch BTX-basiert – einführte, musste der Kunde Transaktionen über Benutzername/Passwort auslösen und mit einer TAN (damals Geldtransaktionsnummern genannt) bestätigen [Bor15]. Das Wissen über den Benutzernamen, und vor allem über das geheime Passwort, war dabei der erste Faktor, während der Besitz der physischen TAN-Liste den zweiten Faktor darstellte.

Die TAN-Liste blieb lange der unbestrittene zweite Faktor. Erst als sich Phishing-Angriffe mehrten, wurde das Verfahren bezüglich seiner Sicherheit in Frage gestellt und 2005 zunehmend durch die indizierte TAN-Liste (iTAN) abgelöst [Gre05]. Bei dem iTAN-Verfahren kann nicht mehr länger eine beliebige Transaktionsnummer der TAN-Liste zur Transaktionsbestätigung genutzt werden, sondern nur noch eine bestimmte, die vom Kreditinstitut vorgegeben wird. Wie schon bei der klassischen TAN-Liste ist eine iTAN nach ihrer Verwendung verbraucht und kann danach nicht mehr genutzt werden.

Die indizierte TAN-Liste konnte gegenüber herkömmlichen TANs die Anzahl der Pishing-Angriffe merklich reduzieren, sorgte seitens der Angreifer jedoch dafür, dass neue Schadsoftware entwickelt wurde, die Transaktionen auf den Rechnern der Nutzer manipuliert — noch vor Eingabe der TAN [Bac09]. Im Folgenden wurden deshalb Verfahren eingeführt, die eine TAN fest an Empfänger und Betrag binden. Ein solches TAN-Verfahren, das seit seiner großflächigen Adaptierung durch die Postbank 2005 auch heute noch Anwendung findet, ist das mTAN-Verfahren. Dabei erhält der Nutzer via SMS eine TAN samt Auftragsdetails auf sein Mobiltelefon. Ferner kann eine Transaktion gemäß Spezifikation nicht auf dem gleichen mobilen Endgerät ausgelöst werden, das die TAN per SMS empfängt. Letztendlich kann das mTAN-Verfahren aber, nicht nur wegen der unverschlüsselten Zustellung der TAN, sondern auch weil Angreifer ein Mobiltelefon und Überweisungsgerät gleichzeitig kapern könnten, nicht als ausreichend starker zweiter Faktor bezeichnet werden [Sch11].

Noch einen Schritt weiter geht daher das auf der CeBIT 2006 vorgestellte chipTAN-Verfahren. Bei dem Verfahren wird mittels eines vertrauenswürdigen TAN-Generators und der persönlichen Bankkarte des Nutzers eine TAN generiert. Nachdem eine Transaktion im Onlinebanking-Portal angestoßen wurde, wird dem Benutzer ein Startcode angezeigt.

Im Folgenden müssen in den TAN-Generator bei eingesteckter Bankkarte der dargestellte Startcode sowie die Überweisungsdetails eingegeben werden. Daraufhin generiert der TAN-Generator eine nur für diesen Auftrag gültige TAN, die der Nutzer zur Transaktionsbestätigung im Onlinebanking-Portal eingeben muss. Obwohl das vorgestellte Verfahren als sicher gilt, bietet es seinen Verwendern wenig Komfort, da neben dem Startcode die Überweisungsdetails doppelt eingegeben werden müssen. Erhöhten Komfort bietet das optische chipTAN-Verfahren, das über Sensoren am TAN-Generator den Startcode und die Überweisungsdetails mittels eines am Bildschirm dargestellten Flickercodes überträgt. Anschließend müssen der am Display des TAN-Generators angezeigte Empfänger und Betrag bestätigt werden, bevor die TAN angezeigt wird. Auch in dieser Ausführung gilt das Verfahren als sicher. Lediglich bei Sammelüberweisungen wurde bisher Missbrauchspotenzial nachgewiesen, da statt den Beträgen der Einzelüberweisungen nur der Gesamtbetrag aller Überweisungen angezeigt wird [Gmb09].

Während das chipTAN-Verfahren hervorragende Sicherheitseigenschaften besitzt und relativ portabel ist, wird sein Komfort oft kritisiert. Mit der zunehmenden Verbreitung von Android- und iOS-Geräten bieten die Sparkassen seit 2014 deswegen ein App-basiertes TAN-Verfahren mit dem Namen pushTAN an, das die TAN in einer eigenständigen App verschlüsselt empfängt und anzeigt. Würde das Verfahren ausschließlich in Verbindung mit eigenständigen Geräten zur Transaktionsauslösung und zum Empfang der TAN genutzt, wäre es in seiner Funktionsweise mit dem mTAN-Verfahren vergleichbar. Da bei pushTAN die TAN verschlüsselt auf das Smartphone übertragen wird, handelt es sich in diesem Fall sogar um einen stärkeren zweiten Faktor als die über SMS unverschlüsselt zugestellte TAN.

Im Gegensatz zum mTAN-Verfahren wird bei pushTAN jedoch *mobiles Onlinebanking auf einem einzigen Gerät* ausdrücklich ermöglicht und gefördert. So kann mit der Banking-App eine Transaktion zuerst ausgelöst und dann mit der TAN-App bequem auf dem selben Gerät empfangen und bestätigt werden. Dazu kommunizieren die beiden Apps in einer Art und Weise, die es dem Nutzer erspart die eingegange TAN abzutippen oder manuell zu kopieren. Als einer der ersten Hersteller App-basierter TAN-Verfahren bieten die Sparkassen ihren Kunden diesen Service bundesweit mit der *S-pushTAN* an. Auch die Hypovereinsbank (*HVB Mobile B@nking*), ING-DiBa (*SmartSecure*) und DKB (*DKB-pushTAN-App*) führten 2014 jeweils ein App-basiertes TAN-Verfahren ein und seit kurzem sind die Volksbanken-Raiffeisenbanken mit *VR-SecureGo* ebenfalls mit einer solchen App vertreten.

In einem Rundschreiben an alle Zahlungsdienstleister der Bundesrepublik Deutschland hat die Bundesanstalt für Finanzdienstleistungsaufsicht (BaFin) im Mai 2015 eine *starke Kundenauthentifizierung* für TAN-Verfahren vorgeschrieben [fF15]. Diese sieht eine Kombination von mindestens zwei verschiedenen Faktoren aus den Kategorien Wissen (z.B. Benutzername und Passwort), Besitz (z.B. eine Smartcard, ein Token oder ein Smartphone) und Inhärenz (biometrische Merkmale) zur Durchführung von Online-Transaktionen vor. Ferner müssen die Faktoren voneinander unabhängig sein, so dass die „Verletzung eines Elements [. . .] keinen Einfluss auf das andere bzw. die anderen“ hat. Obwohl diese Richtlinie in Hinblick auf die etablierten Verfahren eher wie eine Bestandsaufnahme wirkt, da sogar die TAN-Liste diesen Anforderungen gerecht wird, kann ernsthaft angezweifelt

werden, ob App-basierte TAN-Verfahren im mobilen Onlinebanking dieser Bedingung in allen Fällen Rechnung tragen.

3 Eine Sicherheitsanalyse der Sparkassen-Apps

Die beiden Apps *Sparkasse* und *S-pushTAN-App* haben zum Teil ähnliche, zum Teil unterschiedliche Sicherheitskonzepte, die im Folgenden dargestellt werden. Zum Zeitpunkt der Untersuchung war Version 2.7.1 (Build 27269) für die App *Sparkasse* [Gmbb] und Version 1.0.4 (Build 404) für die *S-pushTAN-App* [Gmba] im Google Play Store aktuell.

Inbetriebnahme Während die App *Sparkasse* auf beliebig vielen Geräten installiert und nur mit dem Wissen über die Legitimations-ID und das Passwort verwendet werden kann, muss das pushTAN-Verfahren bei der Sparkasse beantragt und im Anschluss über einen Registrierungsbrief und Erstzugangsdaten freigeschaltet werden. Erst nach Freischaltung kann die *S-pushTAN-App* zur Generierung von TANs verwendet werden. Falls man die App auf mehreren Endgeräten verwenden will, muss jedes Gerät einzeln über einen separaten Registrierungsbrief freigeschalten werden.

PIN Beide Apps verlangen zum Start die Eingabe einer PIN, die bei der initialen Verwendung der App festgelegt wird. Die PIN muss dabei mindestens aus acht Zeichen, einer Zahl, einem Buchstaben und einem Sonderzeichen bestehen. Bei der *S-pushTAN-App* darf die PIN außerdem höchstens fünfmal falsch eingegeben werden bevor die App sämtliche Applikationsdaten löscht. Danach ist eine erneute Inbetriebnahme erforderlich. Die recht restriktive Passwortrichtlinie für beide Apps gibt Grund zur Annahme, dass von vielen Nutzern der Einfachheit wegen die gleiche PIN für beide Apps vergeben wird. Die von uns aufgezeigten Schwächen der *S-pushTAN-App* richten sich aber unabhängig davon nicht gegen die PIN.

Eigene Tastatur Unter Android gibt es seit der Einführung der virtuellen Tastatur die Möglichkeit, die Standardtastatur durch eine Tastatur eines Drittanbieters auszutauschen. Die Verwendung einer solchen Tastatur birgt prinzipiell die Gefahr eines möglichen Keyloggers. Dadurch könnte beispielsweise die PIN, die beim Start der App abgefragt wird, auch ohne Super-User-Rechte ausgespäht werden. Aus diesem Grund liefert die *S-pushTAN-App* eine eigene Tastatur mit, die zur PIN-Eingabe beim Start verwendet wird. Die App *Sparkasse* ist hier weniger restriktiv und verzichtet auf eine eigene Tastatur.

Device-Fingerprinting Um das Kopieren der App auf ein anderes Gerät zu erschweren, implementiert die *S-pushTAN-App* Device-Fingerprinting. Dadurch werden für die Hardware und die installierte Android-Version bestimmte, eindeutige Werte abgefragt und gespeichert. Diese Werte werden bei der Registrierung der App initial abgefragt und abgelegt, um sie bei folgender Verwendung der App wieder abzugleichen. Im Falle von Ungleichheit löscht die Applikation wiederum alle Daten und verlangt eine erneute Inbetriebnahme.

Super-User Während sich die *Sparkassen-App* auch unter `root` mit einem entsprechenden Warnhinweis betreiben lässt, verweigert die *S-pushTAN-App* komplett ihren Dienst, falls die

Umgebung gerootet ist. Dass die *S-pushTAN-App* sich nicht unter `root` betreiben lässt, ist der zentrale Sicherheitsanker dieser App, der es ihr erlauben soll sich gegen Manipulationen durch Schadsoftware zu schützen.

Obfuskierung Obwohl beide Apps mit *ProGuard* obfuskiert wurden, sind sie problemlos dekompilierbar. Das bedeutet insbesondere, dass auf die Implementierung eines eigenen Class-Loaders und auf starke Möglichkeiten zur Verhinderung von Dekompilierung verzichtet wurde. Der Großteil der Klassen- und Methodennamen hat aber durch *ProGuard*s Umbenennungen seine Aussagekraft verloren.

Certificate Pinning Beide Applikationen verwenden konsequent SSL/TLS-geschützte Verbindungen, um das Ausspähen und Modifizieren von Daten durch Dritte zu verhindern. Dennoch kann in solchen Fällen die Verbindung oft durch einen *Man-in-the-Middle*-Angriff (MITM) abgehört werden, indem auf dem Gerät ein Zertifikat für einen MITM-Proxy installiert wird. Um sich gegen solche Angriffe zu schützen, implementieren beide Apps *Certificate Pinning*. Dadurch wird nur bestimmten Zertifikaten das Vertrauen ausgesprochen, so dass ein MITM-Angriff nicht mehr ohne Weiteres möglich ist.

Repacking-Schutz *Repacking* bezeichnet das Dekodieren, Modifizieren, Enkodieren und letztendlich Signieren einer App mit einem neuen Schlüssel. Zur Umgehung von sicherheitsrelevanten Funktionen ist Repacking eine übliche Vorgehensweise, die sich aber auch zum Reverse Engineering eignet. Beide Apps – insbesondere die *S-pushTAN-App* – implementieren Schutzmaßnahmen um Repacking zu verhindern. Zum einen soll somit verhindert werden, dass die App sich trivial modifizieren und verbreiten lässt, zum anderen erschwert dieser Schutz die dynamische Analyse.

Screenreader-Schutz Unter Android gibt es die Möglichkeit sogenannte *Screenreader* zu installieren. Diese dienen der barrierefreien Verwendung des Androidgeräts, bieten aber auch einen legitimen Weg um sensitive Daten ohne `root` auszulesen. Ein Schutz gegen diese Möglichkeit wird nur von der *S-pushTAN-App* implementiert.

PROMON Shield Ein gravierender Unterschied in der Sicherheit der beiden Apps ist die Verwendung einer *Native Library*. Während die *Sparkassen-App* ausschließlich auf Java zurückgreift, verwendet die *S-pushTAN-App* zusätzlich das native *PROMON Shield* des proprietären Anbieters PROMON. Die Verzahnung mit dieser Bibliothek ist in der aktuellen Version der *S-pushTAN-App* äußerst eng. So wird ein Großteil der intern verwendeten Strings in die Bibliothek ausgelagert, die dann über UIDs abgefragt werden. Ferner setzt PROMON Shield statisch-öffentliche, eigentlich konstante Felder im Java-Code über Reflection, um die statische Analyse weiter zu erschweren. Die Bibliothek selbst ist zumindest zum Teil durch einen kryptographischen Schlüssel geschützt, der beim Laden zur eigenen Entschlüsselung verwendet wird. Nicht nur das Reverse Engineering der Bibliothek wird dadurch erschwert, sondern auch das Patchen der Bibliothek erfordert dadurch zusätzliche Schritte.

4 Mögliche Angriffe gegen das App-basierte TAN-Verfahren

Wie zu Beginn erläutert, will die Sparkasse mit dem pushTAN-Verfahren eine Zwei-Faktor-Authentifizierung implementieren, die den Kunden auch bei unachtsamen Umgang mit seinem Zugang schützen soll. Eine klassische Zwei-Faktor-Authentifizierung umfasst aber eine Wissens- und eine Besitzkomponente. Dieser Grundsatz wird mit dem pushTAN-Verfahren gebrochen, da „keine Zusatzgeräte nötig“ sind. Eine Verbindung aus der App *Sparkasse* und *S-pushTAN-App* erlaubt Transaktionen von ein und dem selben Gerät zu veranlassen und zu bestätigen. Aus der *S-pushTAN-App* ist es sogar möglich, die angezeigte TAN direkt in die App *Sparkasse* zu übertragen, ohne diese abzutippen oder zu kopieren. Die starke gegenseitige Integration der Apps lässt die beworbene Entkopplung der Kanäle unserer Auffassung nach nicht erkennen und wirft die Frage auf, warum überhaupt noch zwei Applikationen verwendet werden. Der einzige – jedoch ebenfalls wissensbasierte – Zusatzschutz ist die PIN.

Letztendlich kann insbesondere die Implementierung der *S-pushTAN-App* zwar als technisch stark und hochwertig, deren grundsätzliche Konzeption aber als schwach und angreifbar bewertet werden. Aus diesem Grund ergeben sich vielfältige Angriffsmöglichkeiten, wobei im Folgenden die erfolgsversprechendsten vorgestellt werden.

Angreifermodell und Rahmenbedingungen Das Ziel unserer Demonstration ist ein *technischer Angriff* gegen das App-basierte TAN-Verfahren der Sparkassen. Ein technisch erfolgreicher Angriff erlaubt es einem Angreifer, eine oder mehrere Transaktionen auf technischer Ebene auszulösen oder zu manipulieren. Wird eine Transaktion vom Angreifer ausgeführt, erfolgt dies ohne das Wissen oder den Willen des Opfers. Im Falle einer Transaktionsmanipulation verändert der Angreifer eine vom Opfer wissentlich und willentlich erstellte Transaktion, dessen Auftragsdaten jedoch frei von Fremdeinwirkung sind (kein *Social Engineering*).

Das Angreifermodell basiert auf einem Verwender der *S-pushTAN-App* in Verknüpfung mit der App *Sparkasse* nach dem vorgeschriebenen Nutzungsbild der Sparkasse. Konkret bedeutet dies, dass das Betriebssystem keine Modifikationen, insbesondere kein `root`, aufweist. Jedoch ist für die vom Nutzer verwendete Kombination aus Gerät und Androidversion ein `root`-Exploit bekannt, der es erlaubt das Betriebssystem zu rooten ohne Daten zu verlieren. Dadurch entsteht auch für einen Angreifer die Möglichkeit, einen solchen Exploit auszuführen und entsprechenden Schadcode zu platzieren.

Reverse Engineering der Transaktionsprotokolle Der aus Angreifersicht attraktivste und auch schädlichste Angriff wäre das Reverse Engineering der Transaktionsprotokolle der *S-pushTAN-App* und die Implementierung eines eigenen Clients. Ist dieser Schritt erfolgreich abgeschlossen, müssten nur noch Opfer-spezifische Daten wie Session-Keys, die im Rahmen der Inbetriebnahme erzeugt werden, erlangt werden. Diese Daten könnten durch Kopieren bei physischen Zugriff, durch einen MITM-Angriff oder mittels Instrumentation ausgelesen werden. Da das Reverse Engineering des Protokolls Opfer-unspezifisch und separat erfolgen kann, beschränkt sich der Angriff auf die Daten der *S-pushTAN-App*. Im Anschluss könnte ein Angreifer beliebig viele gültige TANs erzeugen.

Obwohl der Angriff ein hohes Schadenspotenzial aufweist, verlangt er auf der Seite des Angreifers einiges an Entwicklungsaufwand. Das Certificate-Pinning und der Repacking-Schutz verhindern zumindest, dass das Protokoll einfach über einen Man-in-the-Middle-Angriff mitgelesen werden kann. Nicht zuletzt wird das Reverse Engineering durch die getroffenen Obfuskierungsmaßnahmen erschwert. Trotzdem konnten im Rahmen dieser Untersuchung Teile des Protokolls durch App-Interception ausgelesen werden, womit wir betonen wollen, dass die Entwicklung eines eigenen Clients im Bereich des Möglichen liegt.

Kopieren der TAN-App Ein anderer naheliegender Gedanke ist das Klonen der TAN-App inklusive all ihrer Daten. Die Anwendungsdaten sind prinzipiell nur der App selbst zugänglich, weshalb der Angriff `root`-Rechte erfordert. Ein Vorteil des Angriffs wäre, dass die Integrität der App und der Daten vollständig erhalten bleiben kann. Auf der anderen Seite würden für die *S-pushTAN-App* auch Verfahren des Reverse Engineerings erforderlich, um deren Device-Fingerprinting zu verstehen. Die daraus gewonnen Erkenntnisse müssten zur Manipulation am Angreifer-Gerät eingesetzt werden, um die abgeprüften Werte für das Fingerprinting exakt nachzubilden. Neben dem reinen Kopierprozess der App und der Daten, würde also auch ein Skript zum Abfragen bestimmter Geräteeigenschaften notwendig sein. Ferner müsste die PIN erlangt werden. Die einzelnen Schritte des Angriffs könnten sich also wie folgt gestalten:

1. Der Angreifer platziert ein Skript auf dem Gerät des Opfers, das aktiv wird sobald der Nutzer die *S-pushTAN-App* öffnet. Als Resultat liefert das Skript die mitgeschnittene PIN, die App inklusive Daten, sowie die für das Device-Fingerprinting notwendigen Geräteeigenschaften. Je nach gewählter Platzierungsstrategie kann die Zustellung dieser Daten an den Angreifer über das Netzwerk oder auch über USB erfolgen.
2. Das Opfergerät wird nicht länger benötigt und entstandene Spuren sind möglichst zu beseitigen, mindestens in dem Umfang, dass der Eingriff vom Opfer unbemerkt bleibt.
3. Auf dem Angreifergerät muss nun die App sowie die ausgespähten Daten platziert werden. Außerdem muss der Fingerprinting-Algorithmus der App die gleichen Werte lesen wie es auf dem Opfergerät der Fall wäre.

Der eigentliche Realisierungsaufwand für diesen Angriff liegt in dem Verständnis über den Fingerprinting-Algorithmus und welche Werte dieser schlussendlich zur Verarbeitung abfragt. Um an die PIN zu gelangen, scheint nur ein Angriff sinnvoll der sie mitloggt. Ein Brute-Force-Angriff gegen die PIN wird aufgrund der Passwortrichtlinie vermutlich nicht zum Erfolg führen.

Instrumentalisierung beider Apps Während sich die beiden bereits beschriebenen Angriffe darauf konzentrieren würden, dass die Platform – in diesem Fall Android – trotz aller Schutzmaßnahmen nicht als sicher und vertrauenswürdig angenommen werden kann, sollen nun Möglichkeiten beschrieben werden, die das als sicher beworbene Mobilebanking direkt kompromittieren. Zu diesem Zweck werden sowohl die Banking- als auch die TAN-App instrumentalisiert, um entweder Nutzer-induzierte Auftragsdaten zu manipulieren oder

von Seiten des Angreifers beliebige Transaktionen aufzugeben und zu bestätigen. Beide Angriffe haben gemeinsam, dass sie auf das Aktivwerden des Nutzers angewiesen sind, jedoch kein Social Engineering erfordern. Diese Angriffsart wurde letztendlich von uns konkret realisert und stützt unsere Argumentation über die Schwäche App-basierter TANs.

Eine erste Möglichkeit ist die nicht sichtbare Manipulation von Auftragsdaten. Sobald ein Nutzer eine Transaktion in der Banking-App aufgibt, wird diese, kurz bevor sie an den Server des Kreditinstituts übermittelt wird, von der Schadsoftware verändert. Würde im Folgenden dedizierte Hardware wie ein TAN-Generator verwendet, so würde die Manipulation durch die Anzeige des Empfängers und Betrags auffallen. Da die TAN-App aber auf dem gleichen Smartphone verwendet wird, ist es nicht notwendig, den zweiten Faktor gesondert zu kompromittieren. Die Schadsoftware speichert die durch den Nutzer zuvor in der Banking-App eingegeben Daten und manipuliert die Anzeige in der TAN-App derart, dass die erwarteten Werte präsentiert werden. Dadurch kann eine komplette Transaktion mit beliebigen Empfängerdetails ohne das Wissen des Nutzers durchgeführt werden.

Mit entsprechendem Zusatzaufwand könnte der Angriff ausgebaut werden, so dass die Schadsoftware die TAN-App automatisiert verwendet. Zuerst greift die Schadsoftware die Zugangsdaten zum Onlinebanking ab, wenn der Nutzer eine Transaktion ausführt. Danach wechselt der Nutzer in die TAN-App, die er durch Eingabe seiner PIN freigibt. Nun kann die Schadsoftware verhindern, dass sich die TAN-App beim App-Wechsel automatisch wieder sperrt. Im Folgenden könnte der Angreifer nicht nur beliebige Transaktionen anstoßen, sondern diese auch durch den Aufruf der entsprechenden Funktionen der TAN-App automatisiert bestätigen.

5 Der Angriff: Manipulation von Transaktionsdaten

Der von uns letztendlich realisierte Angriff instrumentalisiert und manipuliert die Apps *Sparkasse* und *S-pushTAN-App*. Hierfür wird der Angriff auf einem Gerät platziert, das der Nutzer zum Onlinebanking mit dem pushTAN-Verfahren nutzt. Der Angriff zeigt seine Wirkung sobald der Nutzer eine Überweisung über die App *Sparkasse* auslöst und über die *S-pushTAN-App* bestätigt, indem die vom Nutzer eingegeben Transaktionen durch die des Angreifers ersetzt werden. Dem Nutzer werden jedoch weiterhin in allen Transaktionsschritten beider Apps die von ihm erwarteten Daten präsentiert.

Der Angriff ist als Modul für das Instrumentation-Framework *Xposed* realisiert, das für alle Versionen ab Android 4.0.3 („Ice Cream Sandwich“) verfügbar ist (ausgenommen Android 6). *Xposed* nistet sich tief in Android ein und erlaubt es somit beliebigen Java-Code zu instrumentalisieren. Die Installation von *Xposed* benötigt zwingend `root`, während *Xposed* selbst keine inhärente Bedingung des realisierten Angriffs ist (weshalb er auch eigenständig verwirklicht werden kann). Dennoch bietet *Xposed* für einen *Proof of Concept* eine gute Basis, die uns erlaubt den Entwicklungsaufwand zu reduzieren.

Angriffspunkte der App Sparkasse Die *Sparkassen-App* kann bei der Verwendung des pushTAN-Verfahrens zur Absetzung von Überweisungen genutzt werden und ist deshalb der Ansatzpunkt unseres Angriffs. Grundsätzlich bleibt zu erwähnen, dass die App sich

standardmäßig auch mit `root` verwenden lässt, lediglich ein Hinweis davor warnt, dass dies mit Risiken verbunden sei. In der Eingabemaske zur Erstellung einer Überweisung sind, nach entsprechender Navigation in der App, die folgenden Felder vom Nutzer auszufüllen: Begünstigter, IBAN, BIC, Betrag und Verwendungszweck.

Ziel dieses Angriffs ist es, diese Auftragsdaten zu manipulieren, bevor sie an den Server der Sparkasse gesendet werden, ohne diese Veränderungen zu irgendeinem Zeitpunkt für den Nutzer sichtbar zu machen. Durch Reverse Engineering wurde die Klasse [redacted] und dessen Methode [redacted] als zuständig für das Button-Event Einreichen identifiziert. Durch Instrumentation werden vor der Ausführung dieser Methode die Auftragsdaten des Angreifers vom Server des Angreifers abgerufen und mittels Reflection manipuliert. Nach der Ausführung der Methode werden diese Daten wieder auf die originalen Daten zurückgesetzt und zur späteren Verwendung in der *S-pushTAN-App* gespeichert.

Im Folgenden erscheint ein weiteres Fenster, das nochmals die Auftragsdaten anzeigt und die TAN aus der *S-pushTAN-App* verlangt. Es war im Rahmen des Angriffs nicht nötig, diese Daten erneut anzupassen, da der Server offenbar keine Bestätigung/Kopie der Auftragsdaten sendet oder diese von der *Sparkassen-App* schlichtweg nicht verwendet werden. Im nächsten Schritt muss zur *S-pushTAN-App* gewechselt werden.

Angriffspunkte der S-pushTAN-App Die *S-pushTAN-App* zeigt im Wesentlichen nur die Auftragsdaten und die TAN an. Sie lässt sich allerdings nicht wie die App *Sparkasse* mit `root` betreiben und beendet sich im Gegensatz zu dieser mit einem entsprechenden Warnhinweis. Dieser Schutz musste zunächst deaktiviert werden. Der Hersteller Star-Finanz implementiert selbst keine eigene Überprüfung auf `root`-Zugriff, sondern verwendet hierfür das externe und native Modul *PROMON Shield*. Für *PROMON* gibt es im Java-Code allerdings entsprechende Callbacks, die für den Warnhinweis und das Beenden verwendet werden. Das Callback für die Überprüfung auf `root` in der Methode [redacted] der Klasse [redacted] wurde instrumentalisiert und abgebrochen. Dieses Vorgehen bleibt frei von Konsequenzen, was durchaus überrascht, da die aktuelle Version der *S-pushTAN-App* eine starke Verwebung mit dem *PROMON Shield* aufweist. So sind beispielsweise die Mehrzahl der Zeichenketten in *PROMON* ausgelagert und werden über eine ID abgerufen. Das *PROMON Shield* führt während der Verwendung mehrmals die Methode [redacted] aus, verweigert aber nicht das in ihr ausgelagerte Abrufen von Zeichenketten.

Nach dem Deaktivieren der `root`-Überprüfung lässt sich die App ganz normal verwenden und kann zum Abruf von TANs verwendet werden. Der Server der Sparkasse sendet allerdings die vom Angreifer manipulierten Auftragsdaten und zeigt diese in der *S-pushTAN-App* an. Die angezeigten Daten beschränken sich auf den Betrag und die (maskierte) IBAN. Beide Werte müssen auf die vom Nutzer ursprünglich eingegebenen Werte abgeändert werden. Hierfür stellt der Schadcode die zuvor in der App *Sparkasse* gespeicherten Transaktionsdaten des Opfers wieder her. Zur Manipulation wird die Klasse [redacted] und dessen Methode [redacted] instrumentalisiert. Sie erzeugt aus den vom Sparkassen-Server erhaltenen Daten mehrere Key-Value-Paare zur Anzeige in der App. Dabei werden die beiden Paare Betrag und IBAN entsprechend der abgerufenen Originaldaten des Nutzers abgeändert. Dem Nutzer wird also das erwartete Ergebnis präsentiert, weshalb er die TAN schlussendlich in die *Sparkassen-App* übertragen lässt und den Auftrag somit bestätigt und wirksam macht.

Anwendbarkeit für zukünftige Updates Der demonstrierte Angriff funktioniert nur für die konkreten Versionen der App *Sparkasse* (2.7.1) und *S-pushTAN-App* (1.0.4), für die der Angriff entwickelt wurde. Das liegt darin begründet, dass das vom Obfuskator *ProGuard* eingesetzte Renaming für jeden Kompiliervorgang andere Namen für Klassen und Methoden vergibt. Die Anpassung des Exploits wäre deshalb noch nicht besonders aufwendig und der Angriff könnte sogar derart ausgebaut werden, dass Klassen- und Methodennamen abhängig von der Version nachgeladen werden. Tiefgreifendere Veränderungen an den instrumentalisierten Klassen und/oder Methoden, die bei neuen Versionen ebenfalls zu erwarten sind, würden eine Anpassung des Quelltextes aber zwingend notwendig machen.

Für die *S-pushTAN-App* lässt sich beobachten, dass sie von Version zu Version stärker obfuskiert wird. Das geht in erster Linie auf Verbesserungen des *PROMON Shields* zurück, das immer mehr Teile der Applikation übernimmt und in neuen Versionen besser versucht seine Wirkungsweise zu verschleiern. Mit Veröffentlichung dieser Arbeit wurde zum 16.10.2015 Version 1.0.5 der *S-pushTAN-App* veröffentlicht, die neue Wege bei der Bekämpfung von gerooteten Geräten geht. Diese Version macht unsere bestehenden Lösungsansätze zur Verwendung der *S-pushTAN-App* trotz `root`-Zugriffs zunächst unwirksam, so dass keine unmittelbare Gefahr durch unsere Tools zu erwarten ist. Dennoch möchten wir darauf hinweisen und davor warnen, dass auch diese Version der *S-pushTAN-App* – und alle zukünftigen Versionen – mit entsprechendem Mehraufwand gebrochen werden kann. Stärkeren Maßnahmen zur Code-Obfuskierung kann immer durch stärkeres Reverse Engineering entgegengetreten werden, so dass lediglich der Realisierungsaufwand unseres Verfahrens steigen wird, während die konzeptionellen Schwächen des App-basierten TAN-Verfahrens nie beseitigt werden.

6 Resümee

Der realisierte Angriff zur Manipulation der Auftragsdaten, der sich gegen die App *Sparkasse* und gegen die *S-pushTAN-App* richtet, zeigt deutlich die konzeptionelle Schwäche von App-basieren TAN-Verfahren, wenn auf getrennte Hardware für Transaktionsauslösung und -bestätigung verzichtet wird. Dabei lässt der Angriff es zu keiner Zeit zu, dass die Manipulation durch den Nutzer erkannt wird, weil die angezeigten Daten zu jeder Zeit des Transaktionsprozesses den eingegeben Werten entsprechen. Grundsätzlich könnte der Angriff – mit entsprechendem Mehraufwand – derart erweitert werden, dass er eigenständig auf einem Gerät (ohne `root`, aber mit verfügbaren `root`-Exploit) funktioniert. Da sich der demonstrierte Angriff nicht gegen eine technische sondern gegen eine konzeptionelle Schwäche des Verfahrens richtet, lassen sich App-basierte TAN-Verfahren, und im Speziellen die *S-pushTAN-App*, durch rein technische Nachbesserungen nie vollständig absichern. Für sicheres Onlinebanking empfehlen wir daher, auf den Komfort des mobilen Onlinebankings auf einem einzigen Gerät zu verzichten und Verfahren vorzuziehen, die einen zweiten, vom ersten Faktor unabhängigen Authentifizierungsfaktor darstellen — wie beispielsweise das etablierte chipTAN-Verfahren.

A Screenshots

Abbildung 1 zeigt die Transaktionsmanipulation im Überblick. Nachdem sich ein Nutzer im Onlinebanking anmeldet hat, gibt er eine Überweisung über 0,10 € an das Finanzamt auf, die per TAN bestätigt werden muss. Nach Anmeldung in der pushTAN-App kann der Nutzer die TAN per Click direkt an die Sparkassen-App übertragen. Obwohl die Auftragsdaten auch in der pushTAN-App korrekt zu sein scheinen, zeigt ein Blick in die Umsatzdetails, dass mit dieser TAN tatsächlich 13,37 € an Vincent Haupert überwiesen wurden (statt 0,10 € an das Finanzamt).

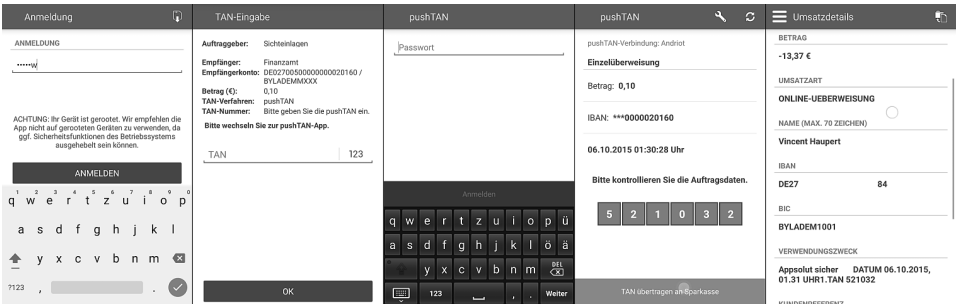


Abb. 1: Die Demo der Transaktionsmanipulation im Überblick.

B Mobile Schadsoftware im Play Store

Wie für jede Schadsoftware stellt sich bei den von uns präsentierten Angriffsmöglichkeiten die Frage, wie Angriffe auf den Geräten der Opfer platziert werden und wie die von uns angenommen Rahmenbedingungen – insbesondere das Erlangen von `root`-Rechten – hergestellt werden können. In einer aktuellen Studie der Universität Cambridge wird gezeigt, dass 87,7% aller Android-Geräte angreifbar gegenüber kritischen Sicherheitslücken sind [TBR15]. Mit sieben von elf untersuchten Verwundbarkeiten können `root`-Rechte ohne physischen Zugang zu dem Gerät erlangt werden. Die Studie zeigt außerdem, dass Sicherheitslücken von Seiten der Hersteller nur langsam, wenn überhaupt, geschlossen werden, was dazu führt, dass viele Geräte auch nach der Entdeckung einer Lücke lange Zeit verwundbar bleiben.

Für die Platzierung des Angriffs sind verschiedene Strategien denkbar. Zum einen könnte die App durch einen Nutzer aus Dritt-Quellen installiert werden, wozu er durch Social-Engineering bewegt werden müsste. Wesentlich gefährlicher ist dagegen das Einschleusen von Schadsoftware in den offiziellen Play Store, das von Google nur unzureichend verhindert wird. Maier, Müller und Protsenko haben 2014 bereits gezeigt, dass sogenannte *Split-Personality-Malware*, also das Aufteilen von Schadsoftware in mehrere Teile, die für sich genommen jeweils unschädlich sind, nicht vom automatisierten Review-Prozess des Play Stores entdeckt wird [MMP14].

Dass schadhafte Apps, die im Google Play Store verfügbar sind und Geräte zunächst rooten bevor sie beliebigen Schadcode zur Ausführung bringen, keine Fiktion sind, hat

im September 2015 die App “Brain Test” gezeigt [PB15]. Diese App, welche heute als besonders gefährlich eingestuft wird, war lange Zeit im Play Store verfügbar und hat auf den Geräten ihrer Nutzer verschiedene `root`-Exploits zur Ausführung gebracht, um im Anschluss daran Schadsoftware nachzuladen und auszuführen. Einer Analyse durch Googles Review-Prozess hat sie sich sich entzogen, indem die automatisierte Analyse-Umgebung erkannt wurde und sich innerhalb dieser Umgebung unauffällig verhalten wurde.

Literaturverzeichnis

[Bac09] Daniel Bachfeld. Mit guten Karten. *c't*, 19:92–95, 2009.

[Bor15] Detlef Borchers. Vor 30 Jahren: Online-Banking startet in Deutschland, 11 2015.

[e.V15] Bitkom e.V. Online-Banking ist bequem und sicher, 08 2015.

[fF15] Bundesanstalt fr Finanzdienstleistungsaufsicht. Mindestanforderungen an die Sicherheit von Internetzahlungen, 05 2015.

[Fre15] Harald Freiberger. Betrugsserie beim Online-Banking, Oktober 2015.

[Gmba] Star Finanz GmbH. S-pushTAN – Android-Apps auf Google Play.

[Gmbb] Star Finanz GmbH. Sparkasse – Android-Apps auf Google Play.

[Gmb09] RedTeam Pentesting GmbH. Man-in-the-Middle-Angriffe auf das chipTAN comfort-Verfahren im Online-Banking, 11 2009.

[Gre05] Detlef Grell. Postbank mit neuem TAN-System gegen Phishing, 08 2005.

[MMP14] Damonik Maier, Tilo Müller und Mykolai Protsenko. Divide-and-Conquer: Why Android Malware Cannot Be Stopped. In *Availability, Reliability and Security (ARES), 2014 Ninth International Conference on*, Seiten 30–39, Sept 2014.

[PB15] Andrey Polkovnichenko und Alon Boxiner. BrainTest â A New Level of Sophistication in Mobile Malware, 09 2015.

[Sch11] Jrgen Schmidt. Angriffe auf deutsche mTAN-Banking-User, 04 2011.

[TBR15] Daniel R. Thomas, Alastair R. Beresford und Andrew Rice. Security Metrics for the Android Ecosystem. In *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices*, SPSM '15, Seiten 87–98, New York, NY, USA, 2015. ACM.

Danksagung: Diese Arbeit entstand im Rahmen des vom Bayerischen Staatsministerium für Bildung und Wissenschaft geförderten Bayerischen Forschungsverbunds “Sicherheit hochgradig vernetzter IT-Systeme” (FORSEC) und dem von der Deutschen Forschungsgemeinschaft geförderten Sonderforschungsbereich/Transregio 89 “Invasive Computing” (InvasIC).

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 113

Die Bordkarte als Authentifikationsmittel bei Flugreisen

Mariia Astrakhantceva[1], Bettina Hübscher[2], Günter Karjoth[3]

Abstract: Mobile Technologien und das Web sind omnipräsent. Auch die Flugfahrtbranche setzt auf die Buchung über das Internet. Diese Angebote erhöhen nicht nur die Benutzerfreundlichkeit, optimieren den Abflugprozess und ersparen Millionen Euro an Kosten, sondern enthalten auch Gefahren – für Passagiere wie Fluglinien. Veröffentlicht jemand das Foto einer Bordkarte auf dem Internet, kann die darauf abgedruckte Information verwendet werden, um beispielsweise den Rückflug zu annulieren. Diese Gefahr ist vielen Passagieren nicht bewußt. Die Möglichkeit dieses Angriffs besteht nicht nur durch das Wissen über die Information auf der Bordkarte, sondern auch durch eine schwache Authentifizierung auf Seite der Fluglinie. Dieser Artikel beschreibt die technischen Grundlagen, diskutiert die rechtlichen Aspekte und erörtert mögliche Gegenmaßnahmen.

Keywords: Sicherheit, Bordkarte, Authentifikation

1 Einleitung

PINs und Passwörter sind unsicher, nervig und schwer zu merken. Inzwischen ist fast jedem bewusst, dass man sein Passwort nicht auf einem Post-it an den Bildschirm seines PCs anheften oder auf andere Weise öffentlich zugänglich machen sollte. Dass dies in anderen Situationen immer noch geschieht, zeigen zahlreiche Bilder von Bordkarten auf Facebook oder Instagram. Dass dieses Posting anderen die Möglichkeit gibt, sich auf der Webseite der Fluggesellschaft im Namen des Passagiers einzuloggen, wurde vor kurzem im Blog "Krebs on Security" beschrieben [Kr15]. Ein Leser dieses Blogs konnte nur mit Hilfe eines Bildes der Bordkarte eines Freundes sich bei der Lufthansa einloggen und dort nicht nur den Flug, sondern auch weitere Flüge, die unter seiner Vielfliegernummer gebucht worden sind, einsehen. Desweiteren hätte er auch den Sitzplan ändern oder diesen Flug annulieren können.

Häufig reichen Nachname und Buchungsnummer respektive E-Ticketnummer aus, um bei Fluglinien als auch Touristikunternehmen wie *Checkmytrip* oder *Viewtrip* Flüge zu verwalten. Hat man Zugriff auf das Buchungssystem bekommen, kann man den Sitzplatz wählen oder wechseln, die Bordkarte erneut ausdrucken, Gepäck hinzufügen, den Flug ändern oder sogar annulieren oder eine Bestätigungs-Email erneut senden.

In diesem Artikel untersuchen wir, wie und auf welche Informationen nur durch das Bild einer Bordkarte zugegriffen werden kann und welche Konsequenzen damit verbunden sind – für Sicherheit wie Datenschutz. Wir beleuchten die rechtliche Situation und erörtern technische wie administrative Maßnahmen, um diesen Missbrauch zu erschweren.

[1] Hochschule Luzern – Wirtschaft, Zentralstrasse 9, CH 6002 Luzern, mariia.astrakhantceva@hslu.ch
[2] Hochschule Luzern – Wirtschaft, Frankenstrasse 9, CH 6002 Luzern, bettina.huebscher@hslu.ch
[3] Hochschule Luzern – Wirtschaft, Zentralstrasse 9, CH 6002 Luzern, guenter.karjoth@hslu.ch

2 Der Abflugprozess

Der Ablauf eines Verkehrsflugs von der Buchung bis zum Einsteigen in das Flugzeug ist in Abbildung 1 schematisch dargestellt. Um das stetig ansteigende Personenaufkommen zu bewältigen, hat 2007 die IATA das Simplifying Passenger Travel (SPT) Programm vorgeschlagen [Gu09], dessen Ziel die Optimierung und Automatisierung des Personenverkehrs unter der Berücksichtigung der Sicherheitsvorschriften ist. Es empfiehlt den Einsatz von Self-Service Kiosks, Web Check-in und automatisierten Bordkartenkontrollen.

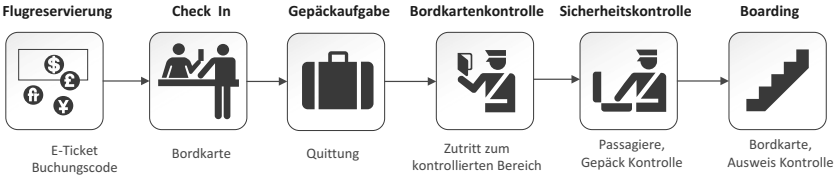


Abb. 1: Ablauf eines Verkehrsflugs von der Buchung bis zum Einstieg

2.1 Buchung

Um ein Flugticket zu kaufen, wendet man sich entweder an ein Reisebüro, besucht die Webseite der Fluglinie oder einer Buchungsplattform. In einem Online-Buchungsprozess öffnet man zuerst eine Buchungsseite, wählt dort den Flug aus und gibt danach seine Personalien an, welche von Buchungsseite zu Buchungsseite ganz unterschiedlich sein können. Die Website *Seat24*[4] verlangt Geschlecht, Vor- und Nachname, Email-Adresse und Handynummer. Sobald der Buchungsprozess abgeschlossen ist, speichert das jeweilige Computerreservierungssystem diese Buchung mit einer eindeutigen Kennung – der Buchungs- oder Reservierungsnummer –, und der Nummer des E-Tickets, welche Passagiere per Email (bei einer Online-Buchung) oder in ausgedruckter Form (bei der Reservierung im Reisebüro) bekommen.

Computerreservierungssysteme verwalten Informationen über Fluglinien, Flugpläne, Verfügbarkeit und Tarife. Sie ermöglichen Reservierungen und unterstützen die Ticketausstellung. Weltweit haben sich vier Systeme etabliert: Amadeus, Galileo, Sabre und Worldspan. Jährlich werden mehr als 400 Millionen Tickets bearbeitet. Die Buchungsnummer kann man sich als einen Schlüssel zur Verwaltung einer Buchung vorstellen. Diese Nummer besteht immer aus einer Kombination von sechs Zahlen und Großbuchstaben, z.B. `AHY6HO`. Es ist wichtig zu wissen, dass der Buchungscode zur Buchung und nicht zum Flug gehört. D.h., wenn innerhalb einer Buchung 10 Flugtickets für fünf Personen gekauft werden, wird nur ein Buchungscode erstellt, mit dem man die ganze Buchung verwalten kann. Die Nummer des E-Tickets ist im Vergleich zum Buchungscode länger und besteht aus 13 Zahlen. Die ersten drei Zahlen sind der IATA-Prefixcode der jeweiligen Fluglinie. Danach folgt die vier-stellige Ticketnummer und die sechs-stellige Seriennummer.

[4] www.seat24.de

2.2 Check-in

Bei vielen Fluglinien können Fluggäste den Check-in auf deren Website selber durchführen (*Web Check-in* oder *Mobile Check-in*) und dabei auch Sitzplätze und Speisen auswählen. Sie erhalten – papierlos – eine Bordkarte, die mit einem Barcode versehen ist, mittels elektronischer Kurzmitteilung oder Email zugeschickt. Bei Bedarf kann diese auch auf dem eigenen Drucker – zu Hause wie im Hotel – ausdruckt werden (Homeprint). Dieser Dienst wird sehr gut angenommen, obwohl Kosten beim Konsumenten anfallen (Smartphone, Drucker, usw.). Die Fluggäste ersparen sich im Gegenzug einen Teil der Wartezeit am Flughafen [Lu15, S. 61f]. Die Gepäckabgabe erfolgt anschliessend am "Bag-Drop" Schalter im Flughafen. Kunden ohne Internetzugang checken sich am Check-in-Schalter oder an einem Automaten ein [Fl15].

Sollte der Akku des Mobiltelefons leer sein, oder man hat sein Mobiltelefon vergessen, so kann man sich jederzeit an einem Check-in-Automaten, am Gepäckabgabeschalter oder am Check-in-Schalter am Flughafen eine Papierbordkarte ausdrucken (lassen).

2.3 Zugang zum kontrollierten Bereich des Flughafen

Flughäfen sind in verschiedene Sicherheitsbereiche unterteilt, deren Zutritt unterschiedlich kontrolliert wird. Der Besitz einer gültigen Bordkarte gibt Zugang zum kontrollierten Bereich des Flughafen. Die Bordkartenkontrolle werden durch automatische Anlagen durchgeführt, die aus einer Sensorschleuse mit automatischen Schwenkflügeln bestehen. Ein integrierter Bordkartenleser erfasst Barcodes von der Bordkarte, dem Homeprint oder auch dem Handy. Die Sensorik sorgt für Einzelzugang der Passagiere. Bei der darauffolgenden Sicherheitskontrolle wird nur die Person und das Gepäck, aber nicht mehr die Bordkarte kontrolliert.

Bei der Verwendung zweier identischer Bordkarten wird der Zugang beim zweiten Versuch gesperrt.[5] Wenn man vor dem originären Besitzer der Bordkarte den kontrollierten Bereich des Flughafens betritt, ist es damit möglich sich ohne Identitätskontrolle, aber nicht ohne Sicherheitskontrolle, im Flughafen zu bewegen.

3 E-Ticket und Bordkarte

260 oder 83% aller Fluglinien sind Mitglieder des Internationalen Luftverkehrs-Verbandes (IATA). Eine Aufgabe der IATA ist es, die Prozesse im Luftfahrtgeschäft zu vereinfachen. Im Bereich der Fluggastbeförderung betrifft dies auch die Vereinheitlichung der Tickets, so dass der Passagier mit einer einzigen Buchung problemlos mit mehreren Fluglinien reisen kann. Im Rahmen des "Simplifying the Business (StB)" Programm stellten alle IATA Mitglieder bis zum 1. Juni 2008 auf ein elektronisches Ticketsystem um. Neben einer hoher Kostenersparnis (ca. 3 Milliarden $ pro Jahr), brachte das E-Ticket auch eine Erleicherung für Passagiere bei Ticketverlust oder Änderung ihrer Buchung.

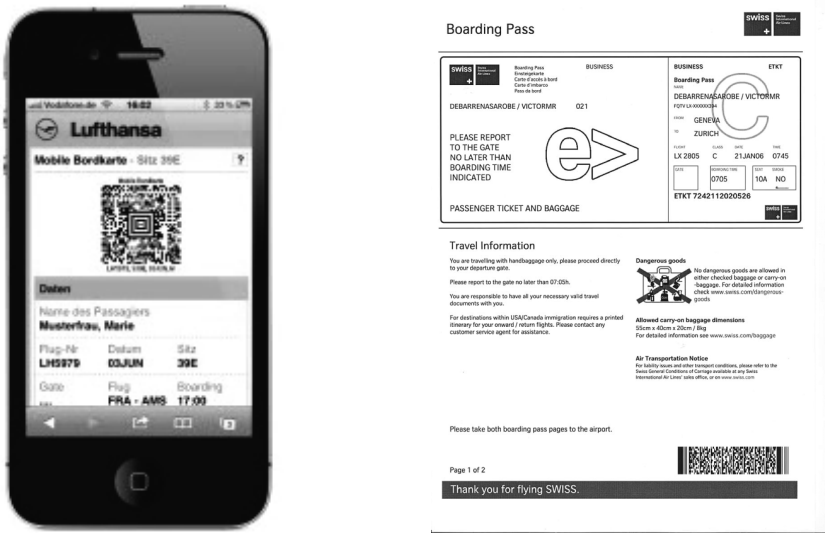
[5] Gemäß Auskunft vom Flughafen Zürich vom 13. September 2015. Siehe aber Kapitel 5.2.

3.1 Bordkarte

Der obligatorische Text und die Grösse einer Bordkarte sind in der IATA Resolution 722c geregelt [IA15b]. Eine Bordkarte im ABT Format enthält mindestens die folgenden Felder: Fluglinie, Name des Passagiers, Sequenznummer, Abflugort, Ankunftsort, Flugklasse, Tickettyp, Flugnummer, Datum des Flugs, Abflugzeit, Boardingzeit, Sitzplatznummer und das Gate.

Abb. 2: Bordkarte im ABT Format [IA15b]

In der Regel stehen weiter die Nummer des E-Tickets, ein Strichcode/QR-Code und oft der Buchungscode auf der Bordkarte. Diese Daten dienen zur Registrierung und Kontrolle im Flughafensystem des Passagiers vom Check-in bis zum Boarding.



(a) Mobile Bordkarte mit QR-Code [Lu15] (b) Ausgedruckte Bordkarte mit Strichcode [IA15b]

Abb. 3: Elektronisch übermittelte Bordkarten

Neben der am Check-in Schalter erhaltenen Bordkarte (Abbildung 2) sieht man immer öfter mobile Bordkarten auf Smartphones oder Smartwatches (Abbildung 3(a)) oder die zu Hause ausgedruckte Bordkarte (Abbildung 3(b)). Beide Kartentypen enthalten einen

Strichcode oder QR Code. Vor- und Nachteile einer mobilen Bordkarte werden in [Go09, SI15] diskutiert.

3.2 Strichcode und QR-Code

Die im Strichcode gespeicherte Information regelt die IATA Resolution 792 [IA10]. Gemäß dieser Resolution enthält der Strichcode neben der bereits aufgeführten Information noch die Nummer des E-Tickets, den Buchungscode und den Status des Passagiers. Desweiteren sind verschiedene Sicherheitsinformationen enthalten.

Mit der App Barcode Reader pdf417[6] haben wir den Strichcode zweier uns zur Verfügung stehender Bordkarten ausgelesen. Es zeigte sich, dass die Information im folgenden Format im Strichcode gespeichert ist (zum Datenschutz haben wir Name, Buchungscode und die Vielflieger-Nummer geändert):

```
M1MUSTERMANN/HANSMR ELKM5HKM HYDBOM9W 452 039H016A0033
                    10F>10B1WW5039B9W
```

Diese Information lässt sich leicht in die folgenden Bestandteile auflösen, wie in Tabelle 1 gezeigt. Wenn der Passagier seine Vielfliegernummer der Fluglinie angegeben hat, ist diese Nummer im Strichcode ebenfalls gespeichert.

Information	Beschreibung
M1	leg without conditional data
MUSTERMANN/HANS	Nach- und Vorname des Passagiers
MR	Geschlecht des Passagiers
E	Kennzeichnung E-Ticket
LKM5HK	Buchungscode
HYDBOM	Abflug in Hyderabad (HYD), Ankunft in Mumbai (BOM)
9W 452	Flugnummer
039	Julianisches Datum des Fluges – in diesem Fall der 8. Februar
0033	Sequenznummer des Passagiers – in diesem Fall war der Passagier die 33. Person, welche sich für den Flug eingecheckt hat
10F>10B1WW5039B9W	spezifische Information für die Fluglinie

Tab. 1: Die im Strichcode gespeicherte Information

Der QR-Code der mobilen Bordkarte enthält noch mehr Informationen als der Strichcode. Um das zu bestätigen, haben wir den gültigen QR-Code einer mobilen Bordkarte von Swiss Airline genommen und mit Hilfe der Internetseite Zxing[7] dekodiert:

```
M1MUSTERMANN/HANSMR EAA2BBS ZRHSFOLX 0038 253Y000A0000 03A>10B0 ILX
                    297421234567891 0LX LX 123456789123456 1PC
```

[6] pdf417.mobi
[7] https://zxing.org/w/decode.jspx

Wie man aus der dekodierten Information sehen kann, kommen im QR-Code die Nummer des E-Tickets (`7421234567891`) und die Vielfliegernummer (`123456789123456 `) hinzu.

4 Angreiferschemen und Konsequenzen

Wie bereits erörtert reichen oft Nachname und Buchungsnummer respektive E-Ticketnummer aus, um bei Fluglinien als auch Touristikunternehmen Flüge verwalten zu können. Bei der Buchung wird diese Information dem Kunden per Email oder Kurznachricht mitgeteilt. Nach dem Check-in ist diese Information auch auf der Bordkarte ersichtlich. Wir stellen uns nun die Frage, wer außerdem Zugriff auf die Bordkarte haben könnte und was mit dieser Information möglich ist.

Bei der Papierbordkarte haben neben dem Passagier häufig Personen aus dem Freundes- und Bekanntenkreis Gelegenheit diese Daten einzusehen. Weitere Personen sind das Reinigungspersonal des Flughafens oder Hotels, welche weggeworfene Bordkarten oder deren Coupons entsorgen. Außerdem bieten viele Hotels die Möglichkeit an, die Bordkarte auf dedizierten und häufig auch für andere leicht zugänglichen Druckern auszudrucken. Mit Angabe der Information übernimmt sogar die Hotelrezeption das Einchecken.

Ein Bild der Bordkarte wird ebenfalls wie beobachtet mit Freunden in sozialen Netzwerken geteilt. Begrenzen die Datenschutzeinstellungen nicht die Sichtbarkeit des Postings, kann auch jeder andere Internetnutzer diese Posts ansehen. Sind Name und E-Ticket Nummer/Buchungscode nicht verdeckt, können diese ohne weitere Hilfsmittel abgelesen werden. Ansonsten reicht es aus den Barcode auf dem Foto zu scannen, um an obige Information zu kommen. Auch wenn auf dem Foto der Barcode nicht gut abgebildet ist, ist die Software der Barcode Scanner häufig in der Lage die Kodierung zu rekonstruieren.

Schlussendlich ist es ebenfalls leicht möglich durch den Kauf eines (billigen) Fluges über das Internet in den Besitz einer Bordkarte zu kommen. Die Identität des Käufers wird dort auf das verwendete Zahlungsmedium reduziert. Wir gehen aber davon aus, dass beim Durchschreiten der automatisierten Kontrolle am Übergang vom öffentlichen in den nicht öffentlichen Bereich eine Videoaufnahme gemacht wird. Bei beiden Arten des Zugangs gilt aber:

> "Wer einen Flugschein erwirbt oder sonstwie in den Besitz eines solchen gelangt ohne die Absicht, die Flugreise auszuführen, ist zum Zutritt mittels dieses Flugscheins nicht berechtigt." [Z14, Art. 3]

Als Berechtigung gelten für Fluggäste die zum Durchschreiten der Zutrittskontrolle und der Passkontrolle notwendigen Ausweise. Diese Ausweise sind nur gültig für den Weg zum und vom Luftfahrzeug.

4.1 Experimente

Wir haben Fotos mit Bordkarten von der Internetseite *WEBSTA*[8] entnommen. Auf vielen war es leicht den Namen des Passagiers, die Nummer des E-Tickets, die Fluglinie und den Buchungscode zu erkennen. Auf vielen dieser Fotos war auch der Strichcode sichtbar. Diese Strichcodes wurden mit der App Pdf417 und der Internetseite Zxing dekodiert. Die ausgelesenen Daten ermöglichten den Zugriff auf die Buchungsverwaltung.

Am Flughafen Zürich stellten wir fest, dass zum Inhalt der Mülleimer nur das Reinigungspersonal Zugriff hat. Aber es ist anzumerken, dass viele Passagiere vor dem Abflug einfach ihre Bordkarte auf dem Tisch im Café liegen lassen. In der Nähe sitzende Personen könnten daher leicht ein Foto der Bordkarte machen oder den Barcode einscannen.

Außerdem wurde im September 2015 ein Flug von der Schweiz nach Dänemark gebucht, den Web Check-in gemacht und geschaut, wie weit man im Flughafen nur mit einer zu Hause ausgedruckten Bordkarte (ohne Kofferabgabe) gehen kann, ohne dass der Ausweis des Passagiers kontrolliert wird. Während des Abflugprozesses wurde die Bordkarte niemals mit dem Ausweis des Passagiers verglichen. Dies bekräftigt, dass auf Reisen im Schengen-Raum Passagiere selten auf ihre Identität geprüft werden.[9]

4.2 Statistik

Um einen Anhaltspunkt zu bekommen, wie viele Bordkarten im Internet veröffentlicht werden, haben wir über einen Monat nach dem Hashtag #boardingpass in *WEBSTA* gesucht. Auf dieser Webseite wurden pro Tag zwischen 50 und 60 mal das Foto einer Bordkarte veröffentlicht, manchmal waren es sogar 100 Postings. In Stichproben verwendeten wir Bordkarten von verschiedenen Fluglinien, um Zugriff auf die Buchungsinformation zu gelangen.

Obwohl die Klassifizierung von Bildern im Allgemeinen als schweres Problem gilt, haben Lorenzi und Vaidya gezeigt, dass es relativ einfach ist Dokumente wie Kreditkarten, Führerscheine und Pässe im Internet zu finden [LV11]. Bilder mit diesen Gegenständen können mit einer hohen Sicherheit (größer 90 %) erkannt werden. Wir gehen davon aus, dass Bordkarten auf Grund ihrer einfachen Form und vorgegebenen Informationsfeldern ebenso leicht zu erkennen sind. Damit wäre es möglich, auch andere Informationsquellen als WEBSTA zu durchforsten.

Wir überprüften die Webseiten von 246 IATA Fluglinien auf die verwendeten Einlogprozedur für die Online Buchungsverwaltung. Wir stellten fest, dass 19% der Fluglinien keine Online-Buchungsverwaltung anbieten. Die am häufigsten verwendete Authentifikationsmethode (50 %) verlässt sich auf die Nennung von Nachname und Buchungscode. Danach

[8] webstа.me

[9] Die Reise wurde vor den Terroranschlägen am 13. November 2015 in Paris unternommen. Am Flughafen EuroAirport in Basel müssen seit dem bis auf weiteres alle Passagiere einen Ausweis beziehungsweise einen Pass beim Einsteige-Gate vorzeigen, bevor sie das Flugzeug betreten.

kommt Name mit Buchungscode oder Name mit Nummer des E-Tickets (26%). Somit kann man bei Dreiviertel aller untersuchten Fluglinien (76%) sich mit Hilfe von der Bordkarte ablesbarer Information einloggen. Nur 5% aller Airlines verwenden eine Einlogprozedur, welche einen weiteren Eingabeparameter benötigt, wie z.B. die Email-Adresse.

5 Mögliche Schäden und Abwehrmaßnahmen

Wird die Buchungsinformation missbräuchlich verwendet, können Schäden sowohl für den Passagier als auch für die Fluglinie und den Flughafen entstehen. Der Passagier erleidet einen Verlust der Privatsphäre,[10] hat möglicherweise zusätzliche Kosten im Falle der Stornierung des Fluges und verliert Zeit bei einer Nachkontrolle. Die Risiken für die Fluglinie beinhalten eine Rufschädigung und Schadenersatzforderungen. Da der kontrollierte Bereich des Flughafens ohne Antritt des Fluges wieder verlassen werden kann, ist ein Einkaufen im Duty Free Shop möglich.

Wie gut ist ein Passagier im Fall eines Missbrauchs rechtlich geschützt? Jede Fluglinie listet auf ihrer Webpräsenz die Rechte und Pflichten sowohl des Passagiers als auch der Fluglinie. Grundsätzlich sollte jeder Passagier diese Bedingungen lesen, um seine Rechte und die Rechte der Fluglinie zu wissen. Wir haben uns die Beförderungsbestimmungen einiger Fluglinien angeschaut. Auf den Webseiten der Lufthansa fanden wir nur diesen Satz, der die Fluggäste über einen sorgfältigen Umgang mit der Bordkarte informiert:

> "Flugscheine sind wertvoll. Sie sind zur sorgfältigen Aufbewahrung und zur Ergreifung der erforderlichen Vorkehrungen gegen Verlust und Diebstahl verpflichtet." [LH14, §3.1.8]

Mit dem Posten eines Fotos der Bordkarte ist diese weder gestohlen noch verloren gegangen. Was gilt, wenn Dritte die Information von der Bordkarte ablesen und missbrauchen?

5.1 Luftbeförderungsvertrag

Durch die Online-Buchung eines Flugtickets, wo die entsprechenden Bedingungen (AGB und weitere Vertragsbestandteile) durch Anklicken eines entsprechenden Buttons Gültigkeit erlangen, ist ein Vertragsverhältnis zwischen dem Flugpassagier und der Fluggesellschaft entstanden. Einzige Schranken bilden das Montrealer Übereinkommen [ICAO], gemäß dem der Luftfrachtführer u.a. für Schäden durch Annullierung einer Personenbeförderung haftet.

In ihren Allgemeinen Beförderungsbedingungen (ABB) weist die Lufthansa darauf hin, dass der Fluggast nur mit Vorweisen eines Flugscheins bzw. elektronischer Buchung an

[10] In einem Fall konnten wir mit Hilfe der Buchungsnummer die komplette Europareise einer arabischen Familie einsehen.

der Beförderungsleistung berechtigt ist. Es wird ausdrücklich festgehalten, dass der Flugschein nicht übertragbar ist. Der Fluggast hat die Pflicht, den Flugschein sorgfältig aufzubewahren und darüber hinaus die erforderlichen Vorkehrungen gegen Verlust und Diebstahl zu treffen. Wenn dies auch auf den Schutz vor Missbrauch zutrifft, stellt sich die Frage „Mit was muss der Konsument rechnen?“ Wenn Verlust und Diebstahl genannt ist, wird wohl auf die physische Entwendung des Tickets hingewiesen und wohl kaum auf die Möglichkeit durch den Erhalt von Angaben auf dem Ticket einen Flug zu stornieren oder oder gar selber in Anspruch zu nehmen.

Kann ein Mitwirken des Fluggastes, aber auch durch Dritte, in Bezug auf einen allfällig entstandenen Schaden bejaht werden, so wird die Haftung seitens der Lufthansa entweder ausgeschlossen oder gemindert. Gilt nun das ins Netz gestellte Foto mit dem Ticket als Mitverschulden oder wäre hier nicht mehr die Pflicht beim Dienstleister, die Kunden auf diesen möglichen Missbrauch hinzuweisen? Die Klärung des Umfangs der Informationspflicht und die Überwälzung der Haftung durch die ABB auf den Kunden ist nicht die einzig zu klärende Frage.

Ebenfalls ist auch der Datenschutz mit seinen Vorschriften zu beachten und es ist zu prüfen, ob diese beim genannten Vorfall eingehalten werden. Einer der wichtigsten Grundsätze für die Bearbeitung von Personendaten ist derjenige der Datensicherheit, die fordert, dass Personendaten durch angemessene technische und organisatorische Maßnahmen gegen unbefugtes Bearbeiten geschützt werden. Damit soll verhindert werden, dass die Personendaten durch (externe) Dritte missbraucht werden können, was im vorliegenden Fall sicherlich nicht erreicht wird. Durch Firewalls kann verhindert werden, dass von außen ungehindert auf Informationssysteme zugegriffen werden kann. Wir setzen dies gleich, dass das Flugunternehmen hier eine sichere Variante des Loginsystems anbieten muss, um einen Datenmissbrauch durch Dritte zu verhindern.

Keine Rolle hierbei spielt, auf welche Art und Weise die Schutzziele bedroht sind. Das kann sich aus einem Fehlverhalten eines Mitarbeiters oder eben wie vorliegend aus einem Hacking-Angriff von Dritten ergeben. Aufgrund der Einfachheit des Systems ist ja gar kein Hacking-Angriff als solcher notwendig. Gemäss der Datenschutzverordnung[11] müssen die organisatorischen und technischen Massnahmen dazu führen, dass Risiken, d.h. Datenschutzverletzungen verhindert werden. Dazu gehört, dass das Unternehmen eine Risikoanalyse (z.B. Risiko einer unbeabsichtigten/unberechtigten Datenweitergabe, -löschung oder -bearbeitung, eines Datenverlustes oder technischen Fehlers, etc.) durchführt und entsprechende Massnahmen in die Wege leitet. Stellt das Unternehmen im Rahmen seiner Abklärungen fest, dass Vorschriften verletzt werden, muss sie Korrekturmassnahmen empfehlen können. Das heisst also, dass die zu erfassende Risikoanalyse durch den Flugbetreiber eben solche möglichen Gefährdungen aufzeigen soll und Maßnahmen daraus abgeleitet und umgesetzt werden.

[11] SR 235.11, Verordnung zum Bundesgesetz über den Datenschutz, www.admin.ch

5.2 Gegenmaßnahmen

Da die rechtliche Verantwortung nicht vollständig geklärt ist, sollten die Fluggäste ihre Bordkarten sorgfältig aufbewahren und insbesondere keine Fotos davon im Web veröffentlichen, gebrauchte Bordkarten oder deren Coupons sicher entsorgen und im Flughafen Bordkarten nicht sichtbar in der Tasche oder Portmonee tragen. Ebenfalls ist es notwendig darauf hinzuweisen, dass Barcodes leicht ausgelesen werden können.

Für Fluglinien wie Flughafenbetreiber ist es wichtig, dass die Passagiere im Rahmen der Flugbuchung ausreichend über die Gefahren informiert werden, die durch eine Veröffentlichung einer Fotografie der Bordkarte im Internet bestehen. Diese Maßnahme ist kostengünstig und kann schnell eingeführt werden, da die bestehende Infrastruktur beim Abflugprozess unverändert bleibt. Wie dies gemacht werden kann, zeigt eine aktuelle Meldung der Schweizer Koordinationsstelle zur Bekämpfung der Internetkriminalität KOBIK, in der sie davor warnt, Fotos von Dokumenten mit persönlichen Daten Dritter im Internet zu veröffentlichen, da diese missbraucht werden können [Ko15].

Eine reaktive Maßnahme ist die Speicherung der IP-Adresse, von der eine Buchungsänderung vorgenommen wird. Es erleichtert die Angreifersuche im Falle eines Mißbrauchs. Die Idee ist nicht neu. Die Fluglinie Lufthansa speichert IP-Adressen, wie Malte Spitz in seinem Buch [Sp14] bestätigt. IP-Adressen und Cookies werden von vielen Webseitenbetreibern gespeichert, wie auch Viewtrip und Checkmytrip in ihren Datenschutzbedingungen erläutern. Jedoch ist es möglich die IP-Adresse zu verschleiern.

Eine aktive Maßnahme ist die Überwachung der Doppelverwendung einer Bordkarte beim Zutritt zum kontrollierten Bereichs des Flughafens, wie bereits in Kapitel 4 beschrieben. Obwohl technisch einfach zu verwirklichen und auch vom Flughafen bestätigt, scheint hier die Tücke im Detail zu bestehen. Im Januar 2016 ist es am Flughafen Zürich einem Paar mit Hilfe der ausgedruckten Bordkarte und der mobilen Bordkarte gelungen gemeinsam mit nur einem Ticket bis an das Gate zu kommen [20min]. Gemäß Auskunft Flughafen lag es an den Barcodes der Fluglinie Air France, da diese bei mobiler und ausgedruckter Bordkarte nicht identisch waren.

Die heute gebräuchliche Einlogprozedur für die Buchungsverwaltung verhindert wie aufgezeigt den Mißbrauch nicht, da sie sich auf Information stützt, die als geheim betrachtet wird – Name und Buchungsnummer/E-Ticketnummer. Es sollte daher eine zusätzliche Information, die nicht auf der Bordkarte vermerkt ist, zur Authentifizierung mitverwendet werden. Diese Maßnahme ist bereits bei einigen Fluglinien wie Easyjet implementiert. Neben der Email-Adresse sind weitere mögliche Eingabeparameter die Nummer der Karte, mit der der Flug bezahlt wurde, oder ein Geheimwort. Eine 2-Faktor-Authentifikation, wie heutzutage im Homebanking allgemein üblich, kann durch die Bindung der Bordkarte an ein Mobiltelefon erreicht werden. Zumindest beim Zutritt zum kontrollierten Bereich des Flughafens wäre genug Zeit, um auf einen durch eine SMS übertragenen PIN warten zu können. Ein Mißbrauch durch den Besitzer der Bordkarte ist zumindest erschwert, da die beteiligten Personen sich vor und hinter der Kontrolle befinden und den PIN erst übermitteln müssten.

Seit mehreren Jahren wird auch an die Verwendung von biometrischer Information gedacht [Pr09]. Führte SAS bereits im Jahr 2008 als erste Fluglinie einen optionalen biometrischen Check-in für Inlandflüge ein [GW09], hat sich dieser Ansatz noch nicht durchgesetzt, da es immer noch Pilotprojekte wie von der Alaska Airlines gibt, in denen untersucht wird, inwieweit ein Fingerabdruck die klassische Bordkarte ersetzen kann [Ai15].

Wird der Buchungscode nicht oder nur teilweise auf der Bordkarte aufgeführt, besteht keine Möglichkeit mehr die Originalität der Bordkarte automatisch zu erkennen. Es besteht aber die Möglichkeit die im Barcode kodierte Information zu verschlüsseln [KK14], was aber dessen Kapazität verringert und eine Anpassung der Lesegeräte erfordert. Ausserdem ist die sichere Verwaltung der notwendigen Schlüssel eine große Herausforderung.

6 Fazit

Buchungsinformationen eines Passagiers sind von Dritten einsehbar und können mißbraucht werden, wenn Fotos von Bordkarten auf dem Internet veröffentlicht oder anderweitig zugreifbar sind. Dies kann sowohl Fluggast wie auch Fluglinie schädigen. Name und Buchungsnummer, die zur Authentifikation meistens verwendet wird, ist nicht nur im Klartext auf der Bordkarte abgedruckt, sondern auch in dem Barcode enthalten. Die dadurch entstehende Gefahr ist vielen Passagieren nicht bewußt. Der Einsatz automatischer Bilderkennungsroutinen erhöht die Gefahr, dass für einen spezifischen Missbrauch die dafür geeignete Bordkarte gefunden werden kann.

Um diese Risiken zu vermindern, gilt es neben einer Aufklärung der Passagiere auch die heute übliche Authentifikation um eine zusätzliche Information, die nicht auf der Bordkarte abgebildet ist, zu erweitern. Ein Verzicht des Aufdruckes von Name und E-Ticketnummer würde wohl die Funktionalität bestehender Systeme beinträchtigen. Der Einsatz biometrischer Information wäre ein starker Eingriff in die informationelle Selbstbestimmung der Passagiere. Auf jeden Fall gilt es eine transparente Sicherheitsanalyse durchzuführen und sich nicht hinter (geheimen) behördlichen Anforderungen zu verschanzen. Die Annahme, dass die Nummer des E-Tickets als ein geheimes Passwort geeignet ist, hat sich als Fehleinschätzung herausgestellt.

Danksagung

Die Autoren danken Konrad Marfurt und den Gutachtern für ihre hilfreichen Kommentare.

Literaturverzeichnis

[20min] Flughafen Zürich: «Wir konnten mit einem Ticket zu zweit ans Gate» 28. Januar 2016 05:51; Akt: 28.01.2016 09:56, www.20min.ch/schweiz/zuerich/story/-Wir-konnten-mit-einem-Ticket-zu-zweit-ans-Gate–16718494.

[Ai15] Fingerabdruck statt Bordkarte: Alaska Airlines prüft neue Technik, www.airliners.de/fingerabdruck-bordkarte-alaska-airlines-technik/36405.

[LH14] Lufthansa: Allgemeine Geschäftsbedingungen – Beförderungsbedingungen für Fluggäste und Gepäck (ABB Flugpassage), www.lufthansa.com/online/portal/lh/cmn/generalinfo?nodeid=1761532.

[Fl15] Check-in Methoden, www.flughafen-zuerich.ch/passagiere-und-besucher/abflug-ankunft/check-in-methoden.

[GW09] Glambedakis, A., Watson, G.: Pervasive computers in aviation passenger risk profiling. In *Risk Assessment and Management in Pervasive Computing : Operational, Legal, Ethical, and Financial Perspectives*, pp. 246–261, 2009.

[Go09] Flugticket fürs Mobiltelefon: schneller, besser, sicherer?, www.zdnet.de/41002453/flugticket-fuers-mobiltelefon-schneller-besser-sicherer/4.

[Gu09] Gupta, A., Davidson R.: Simplifying passenger travel (SPT) program. In: Third symposium and Exhibition on ICAO MRTDs, Biometrics and Security Standards. 2009.

[IA10] Passenger Services Conference Resolutions Manual, www.iata.org/whatwedo/stb/documents/resolution792-june2010.pdf.

[IA15b] Simplifying the Business: Bar Coded Boarding Pass Implementation Guide, www.iata.org/whatwedo/stb/documents/bcbp_implementation_guidev4_jun2009.pdf.

[Ko15] Koordinationsstelle zur Bekämpfung der Internetkriminalität, KOBIK: Achtung! Veröffentlichen Sie im Internet keine Fotos von Dokumenten, die persönliche Daten Dritter enthalten. https://www.cybercrime.admin.ch/kobik/de/home/warnmeldungen/2015/2015-09-22.html, September 2015.

[ICAO] ICAO: Übereinkommen zur Vereinheitlichung bestimmter Vorschriften über die Beförderung im internationalen Luftverkehr (kurz: Montrealer Übereinkommen), 1999. https://www.admin.ch/opc/de/official-compilation/2005/4395.pdf

[KK14] Krombholz, K.; Frühwirt, P.; Kieseberg, P.; Kapsalis, I.; Huber, M.; Weippl, E.: QR Code Security: A Survey of Attacks and Challenges for Usable Security. Human Aspects of Information Security, Privacy, and Trust. pp. 79–90, 2014.

[Kr15] Krebs, B.: What is in a Boarding Pass Barcode? A Lot. Krebs on Security, 2015. krebsonsecurity.com/2015/10/whats-in-a-boarding-pass-barcode-a-lot.

[Lu15] Die Mobile Bordkarte - mobil einchecken und Zeit sparen, www.lufthansa.com/ch/de/Die-Mobile-Bordkarte-auf-vielen-Strecken-weltweit-verfuegbar.

[LV11] Lorenzi, D.; Vaidya, J.: Identifying a Critical Threat to Privacy Through Automatic Image Classification. In: First ACM Conference on Data and Application Security and Privacy (CODASPY '11). ACM, pp. 157–168, 2011.

[Pr09] Pravir K. Chawdhry, Rui Pareira Da Silva: Advanced registered traveler paradigm using dynamic risk profile and multimodal biometrics. In: Systems, Man and Cybernetics. pp. 3946–3951, 2009.

[SI15] Mobile evolution, www.sita.aero/resources/air-transport-it-review/air-transport-it-review—issue-3-2014/mobile-evolution—beacons -biometrics-wearables.

[Sp14] Spitz, M.: Was macht ihr mit meinen Daten? Hoffmann und Campe, 2014.

[Z14] Flughafen Zürich: Betriebsreglement für den Flughafen Zürich vom 30.6.2011, Anhang 3: Zutrittsordnung für das nichtöffentliche Flughafengebiet. Stand am 1. Juli 2014

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 125

Correlation-resistant Fuzzy Vault for Fingerprints

Johannes Merkle[1], Moazzam Butt[2], Ulrike Korte[3], Christoph Busch[4]

Abstract: The fuzzy vault is one of the most popular biometric encryption schemes for protecting fingerprint data. However, its implementation faces two challenges: First, the fingerprints need to be aligned. Some publications have proposed the storage of auxiliary data to assist alignment, but these data may leak information about the biometric features. Secondly, the fuzzy vault is susceptible to attacks that correlate the data from two protected templates, which does not only violate the requirement of unlinkability but also allows the recovery of the biometric data.

In this work, we present a fuzzy vault construction for fingerprint data (minutiae) that addresses both issues. We do so by applying an absolute alignment method to the fingerprints, performing a quantization of the minutiae positions to a grid, and using all grid points unoccupied by minutiae as chaff. This approach results in all vaults containing the same set of points. In order to improve recognition performance, we also use the minutiae's angles and types. We present experimental evaluations and compare the results with the existing works on fuzzy fingerprint vault.

Keywords: Fuzzy Vault, Biometric Template Protection, Fingerprint Recognition

1 Introduction

Biometrics is increasingly used for secure identification throughout the world. However, the storage of biometric reference data raises privacy concerns and, thus, demands rigorous protection. While in password-based authentication, typically, one-way hash functions are used to protect the reference data of the user's passwords, this approach cannot be easily adopted for biometric data due to the noise inherent in its measurement. *Biometric template protection* aims to solve this issue by combining cryptography with error tolerance or error correction techniques: the biometric reference data are stored as protected templates, which preserve the privacy of the biometric information, but still allow biometric verification without the need to maintain secret keys for decryption.

For template protection, the following privacy properties are required [CS09, IS11]. Firstly, the protected templates prevent the recovery of the original biometric data unless a sufficiently similar feature data is provided for comparison (*Irreversibility*). Secondly, many different protected templates can be generated from the same biometric data (*Revocability*) so that it is impossible to link two protected templates generated from the same biometric data (*Unlinkability*).

[1] secunet Security Networks AG, Eschborn, Germany
[2] Fraunhofer-Institute for Computer Graphics Research IGD, Darmstadt, Germany
[3] Bundesamt für Sicherheit in der Informationstechnik, Bonn, Germany
[4] da/sec - Biometrics and Internet Security Research Group, Hochschule Darmstadt, Germany

One of the most popular template protection schemes is the fuzzy vault [JS02], which works on unordered feature sets and is, thus, considered particularly eligible for protecting data representing fingerprint minutiae (endings and bifurcations of the skin ridges). In the protected templates (*vaults*) generated by the fuzzy vault, the elements of the feature set are represented as finite field elements and then evaluated on a random secret polynomial of small degree; the resulting set of points on the polynomial's function curve represent a redundant encoding of the polynomial and ensure the error tolerance in case that some of the minutiae are not detected or if additional minutiae are measured during verification. Finally, these *genuine points* are hidden by adding a large number of random *chaff points* not lying on the polynomial's function curve.

However, there are two challenges for the application of the fuzzy vault to fingerprints: First, fingerprints are typically not aligned to each other, i.e., different captures of the same finger are rotated and shifted with respect to each other. Some publications propose to store auxiliary data (extracted from the fingerprint's orientation field) along with the vault to aid alignment [UJ06, NJP07, NNJ10], but this data may leak biometric information. Secondly, the fuzzy vault is susceptible to correlation attacks: Since the chaff points are chosen at random, the intersection of two vaults of the same fingerprint is likely to contain most of the genuine points, while most chaff points will be filtered out; this does not only allow the correlation of two vaults of the same subject (violating the unlinkability requirement) but can even allow recovery of the biometric data [SB07, KY08].

In this work, we present a fuzzy vault construction for fingerprints, which tackles both issues. Using the directed reference point estimation method from [Ta13a], we represent the minutiae's position and angle relatively to a coordinate system that can be robustly computed from the fingerprint. This approach is equivalent to a pre-alignment of the fingerprints [Ma09] and eliminates the need to store auxiliary alignment data. Furthermore, we quantize the minutiae positions to a grid and use all remaining grid points as chaff points. This implies that each vault contains the same set of points, which thwarts the aforementioned correlation attacks. Besides the minutiae's positions, we also use their angles and types, but since they may reveal too much information [CS09],[4] we use them to obscure the outputs of the secret polynomial. Furthermore, we store remainders of the minutiae angles (resulting from a modulo operation with a quantization parameter) to allow some error correction of the minutiae's angles.

Section 2 gives an overview of previous work. In Section 3, we describe our fuzzy fingerprint vault in detail and discuss its security in Section 4. In Section 5, we present the results of experimental evaluations, and compare the observed error rates and security estimates with that of comparable constructions. Finally, Section 6 gives a conclusion.

[4] Minutiae angles resemble the direction of the fingerprint's orientation field, and chaff points, in order to be indistinguishable from genuine points, would also need to have angles that are in accordance with it.

2 Previous Work

The first fuzzy fingerprint vault constructions in [CKL03] and [UPJ05] used only the locations of the minutiae, and assumed the fingerprints to be pre-aligned. In [UJ06], these ideas were improved by deploying an algorithm for fingerprint pre-alignment based on high curvature points as auxiliary data, and [NJP07] extended this construction by using also minutiae angles. The fuzzy fingerprint vaults of [Li08] and of [YV05] also used minutiae positions and angles, but deployed alternative pre-alignment methods using auxiliary data based on topological structures around the core and on a reliable reference minutiae, respectively. In contrast, the implementation of [Li10] used minutiae descriptors that are alignment-independent. The multi-finger fuzzy vault of [Me11] used minutiae positions and performed relative alignment using a minutiae matching algorithm.

All constructions mentioned so far are susceptible to the correlation attack that was sketched in [SB07] and practically implemented in [KY08]. The first fuzzy fingerprint vault immune to this attack was presented in [Ta13a]; there the fingerprints were pre-aligned by means of a directed reference point estimation (also used by our construction), the minutiae data (positions and angles) were quantized to a hexagonal grid, and all unoccupied grid points were used as chaff. A slight improvement (deploying the improved fuzzy vault scheme of [Do03]), together with a detailed analysis, was presented in [TMM15]. The implementation of [Ta15a] applies this construction to multiple fingerprints using Reed-Solomon list-decoding. In [Ta15b], another implementation of the improved fuzzy fingerprint vault resistant to correlation attacks was proposed that used quantized minutiae angles and positions, pre-aligned using the method of [Ta13a], and three different, alignment-independent minutiae descriptors.

3 Proposed Fuzzy Vault Construction

In this section, an approach is proposed to include minutiae angles and types in the fuzzy vault in a way that it is still hard to distinguish between genuine and chaff points. The use of minutiae angles and types will increase the amount of information (entropy) per minutiae, which results in higher security against brute force [MMT09] or false accept attacks [TMM15] and improves the recognition performance (as shown in Section 5).

One of the pre-requisite conditions for fuzzy vault decoding is the correct alignment of the query fingerprint minutiae template with the enrolment fingerprint minutiae template. In this work, the alignment has been performed on the fingerprint samples using the tented arch reference point estimation based on the orientation map around the core point [Ta13a]. After the alignment of fingerprint samples, the minutiae are extracted from these pre-aligned fingerprint samples. These minutiae templates are used as enrolment and query minutiae templates in the fuzzy vault encoding (enrolment) and decoding (verification) process, as explained in the following sections. The parameters used have been determined empirically, as explained in Section 5.

The processes of enrolment and verification are shown in Figure 1.

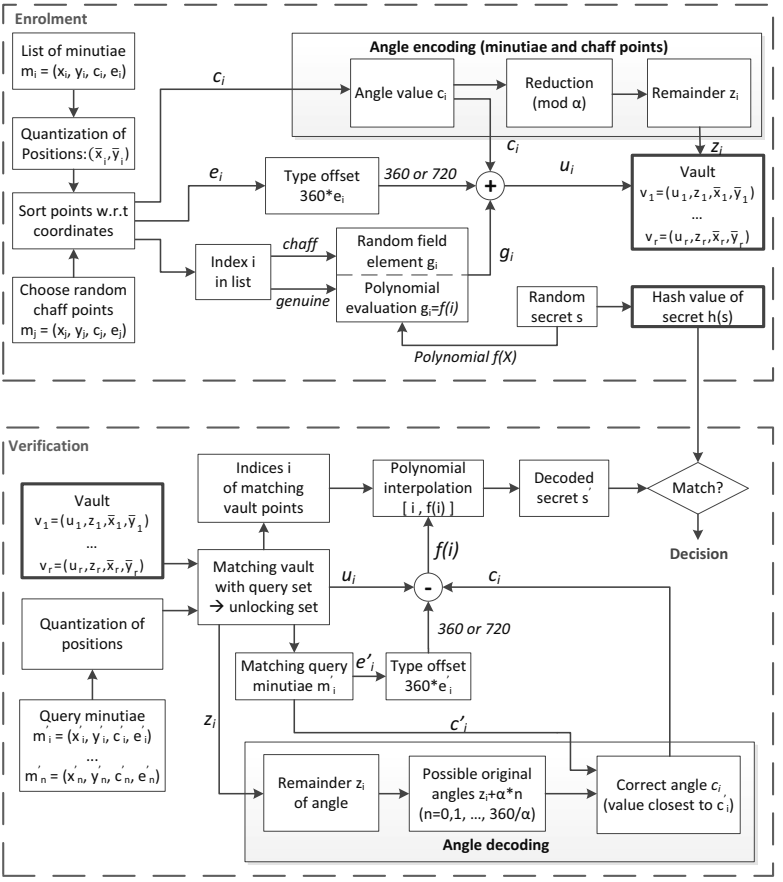


Fig. 1: Enrolment and verification process

3.1 Enrolment

Let the minutiae in the enrolment template be represented as $m_i = (x_i, y_i, c_i, e_i)$, where (x_i, y_i) are the Cartesian coordinates of the position, c_i is the orientation (measured by the angle with the horizontal axis) and e_i the type of the minutia. The minutiae type $e_i = 1$ represents a ridge ending and $e_i = 2$ a ridge bifurcation; minutiae of unknown types ($e_i = 0$) are neglected in our scheme.[5] Furthermore, we discard all minutiae lying outside a region of interest defined by an ellipsis area having semi-minor and semi-major axis lengths (a, b) (see Figure 2).

[5] While further minutiae types can be distinguished (see [Ma09]), commercial minutiae detection algorithms only output these three types as specified by ISO/IEC 19794-2.

The positions (x_i, y_i) of the remaining minutiae points are then quantized according to a rectangular grid with bin spacing q_x and q_y (determined parameters empirically), respectively, as shown in Figure 2, so that the quantized values $(\bar{x}_i, \bar{y}_i)$ are the centers of the corresponding cell in which the minutiae is located. In cases, where two or more minutiae are found in the same cell, the data (quantized position, angle and type) of the minutia having the highest quality value (output by the minutiae extractor) are taken and those of the others are discarded.

From the remaining quantized minutiae, those t having highest minutiae quality value (calculated by the minutiae extractor) are selected for the enrolment process. If less than t quantized minutiae are left over, all of them are chosen.

In order to obscure this set of (at most t) minutiae, all unoccupied (by minutiae) cells of the quantization grid within the ellipsis are added in the minutiae list [Ta13b] as chaff points. Angle and type of these chaff points are randomly generated.

The joined set of r points, comprising the (at most t) quantized minutiae (genuine points) and the chaff points, are sorted with respect to their coordinates. The total number r of points equals the number of grid cells in the region of interest. To protect the vault, a random polynomial f of degree k over a finite field $\mathbf{F}_p$ of prime order $p \geq r$ is chosen; the concatenation of the $k+1$ coefficients represents the binary secret s that must be recovered during verification. For each genuine point, its index i in the sorted list of vault points, represented as finite field element, is evaluated over the polynomial f; the resulting value $f(i)$ is then added to the minutia's angle value c_i and an offset value depending on the minutia's type e_i, resulting in an ordinate value $u_i = f(i) + c_i + 360 \cdot e_i$. For each chaff point, the value u_i is simply chosen as a random finite field element g_i, added to a random angle c_i and an offset $360 \cdot e_i$ with random $e_i \in \{1, 2\}$.

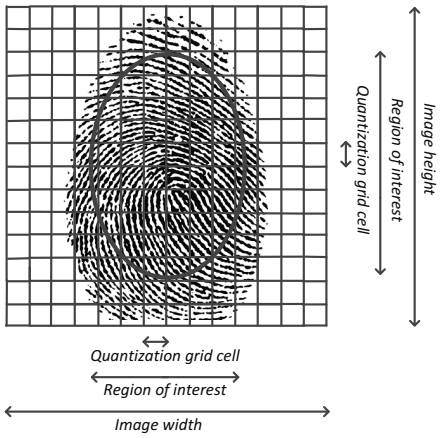


Fig. 2: Fingerprint region of interest and quantization grid

In order to minimize verification errors due to inaccurately measured minutiae angles, we store, for each genuine point, the remainder $c_i \bmod \alpha$, where parameter α is a divisor of 360. For chaff points, the remainder $z_i = c_i \bmod \alpha$ of a randomly chosen angle c_i is stored.

The resulting vault is the list of r points, containing (at most t) genuine points, each of which comprising the values $(u_i, z_i, \bar{x}_i, \bar{y}_i)$. In addition, a hash value $h(s)$ of the secret s is stored in the vault.

3.2 Verification

For verification, a query minutiae template is acquired containing a number of minutiae, each of which represented as $m'_j = (x'_j, y'_j, c'_j, e'_j)$, as described in Section 3.1. As during enrolment, minutiae having unknown type and minutiae located outside the region of interest defined by an ellipsis with semi-minor and semi-major axis lengths (a,b) (see Figure 2) are excluded. The positions of the remaining minutiae are quantized according to a rectangular grid with bin spacing q_x and q_y as shown in Figure 2; the quantized value $\bar{x}'_j, \bar{y}'_j$ is the center of the cell in which the minutia lies. In case two or more minutiae are quantized to the same position, the data (angle and type) of that one having the highest quality value (output by the minutiae extractor) are taken and that of the others are discarded.

Then, for each quantized minutiae position $(\bar{x}'_j, \bar{y}'_j)$ in the query template, the matching point $(u_i, z_i, \bar{x}_i, \bar{y}_i)$ with $(\bar{x}_i, \bar{y}_i) = (\bar{x}'_j, \bar{y}'_j)$ in the vault is identified and added to the list of candidates, referred to as unlocking set. (Since the vault contains all quantized positions in the region of interest, such a point always exists). Then, for each candidate point, the angle c'_j of the mating query minutiae is used to (try to) recover the original angle value c_i as the angle $\bar{c}_i$ that is closest to c'_j and equals z_i up to a multiple of α, i.e., as

$$\bar{c}_i = \operatorname{argmin}_{\beta \in [0,359]} (\Delta(c'_j, \beta) \,|\, \beta = z_i \bmod \alpha),$$

where Δ denotes the distance of two angles along the unit circle. If $\Delta(c_i, c'_j) < \alpha/2$, i.e., if the angle of the query minutiae deviates from the original angle of the mating vault point by less than $\alpha/2$, the original angle c_i is correctly recovered.

For each point in the unlocking set, the recovered angle $\bar{c}_i$ is used to compute an abszissa value $f_i = u_i - \bar{c}_i - e'_i \cdot 360$. If a point is a genuine point, its angle has been correctly recovered, and if its minutia type e_i equals the type e'_j of the mating query minutiae, we obtain $f_i = f(i)$, where i is the index of the point in the vault. However, for chaff points, or if its angle is incorrectly recovered, or if its type deviates from that of the mating query minutia, most likely $f_i \neq f(i)$.[6]

For each combination of $k+1$ points (i, f_i), a supporting polynomial is determined by Lagrange interpolation and checked for correctness by comparing the hash value of its concatenated coefficients with the hash value of the original secret s stored in the vault. If a polynomial is found, where the hash matches, the verification is successful.

[6] Since f_i is the sum of three random variables, there is a small chance that it matches $f(i)$ by incident.

4 Security

A very important aspect of a fuzzy vault implementation is its resistance to recovery attacks, i.e., the effort for an attacker given a vault record to recover the original feature sets or, equivalently, the secret polynomial. When estimating the security of our fuzzy fingerprint vault, we count the expected number of polynomial decoding attempts. This estimate is conservative insofar as we neglect the computational effort for each attempt.

Generally, the fuzzy vault can be attacked by a *brute-force attack*, where the attacker repeatedly samples k from the vault and tries to interpolate the secret polynomial from these. The expected number of attempts until k correct points are chosen can be estimated as $\binom{r}{k}/\binom{t}{k}$ [MMT09]. However, in our implementation, the attacker also has to guess, for all chosen points, the corresponding minutiae angles c_i and types e_i from the given remainders $z_i = c_i \bmod \alpha$, resulting in an additional factor of $(2 \cdot 360/\alpha)^k$.

The above attack can be improved by exploiting the statistical distribution of the feature data. Precisely, the attacker can use estimations for the distributions of minutiae positions and angles, e.g., obtained from public fingerprint databases, to rank the vault points according to the probability of occurrence of the corresponding feature data, and sample the points for polynomial interpolation only from the, say w, top ranked points. Furthermore, instead of guessing random values for the minutiae angles (matching the given remainders z_i) and types, the attacker can choose the most likely values. In order to analyze the success probability of this *statistical attack*, we empirically determined the distribution of the feature data stored by our fuzzy vault, i.e., the distribution of the minutiae positions quantized to the grid and of the remainder (modulo α) of minutiae angles at each grid point, computed from over 130.000 minutiae extracted with the FingerJetFX algorithm from 2500 fingerprints from the MCYT-100 database [OGFAS03]. Using this distribution, we determined, for each grid point, the predominant α-rounded minutiae angles $c - (c \bmod \alpha)$ and minutiae types as the attacker's guess for that position. Then, we estimated, for different $w \geq k$, the probability $P(w)$ that, for a randomly generated vault record, the top ranked (according to the distribution of the minutiae data) w vault points contain at least k genuine points, for which both the minutia's α-rounded angle and type assume the most predominant value. Using these estimates, the expected number of attempts until the attacker is successful can be estimated as $P(w)^{-1}\binom{w}{k}$, minimized over w.

In contrast, the *false-accept attack* exploits the specific distribution of the biometric features simply by simulating repeated (impostor) verification attempts using the features of randomly chosen (real) fingerprints, e.g., chosen from a biometric database [TMM15]. The success probability of the false-accept attack is equal to the False Match Rate (FMR) for the parameters used. In general, the attacker can deviate in her simulation from the parameters used in actual operation to optimize her success rate; however, in our fuzzy vault implementation, the number n of decoding iterations is the only parameter that is not already fixed in the enrolment. It has been proven in [TMM15] that the expected number of decoding attempts of the false-accept attack is minimized for $n = 1$. Hence, we estimate the security against false-accept attacks using this optimal strategy.

Estimating very high security levels assumes sharp estimations of FMRs when they are close to zero. In biometric systems with deterministic verification algorithm, the FMR can only be estimated down to the magnitude of $1/N$, where N is the number of impostor verifications performed in the evaluation. However, the verification of our implementation is probabilistic as soon as the unlocking set contains more than k points. This property allows us to give heuristic estimates of FMRs that are much smaller than $1/N$: For each single impostor verification, we compute the success probability based on the size of the unlocking set and the number of correct points contained, and, finally, we estimate the FMR as the mean over all verifications. For details, we refer to [TMM15].

Another very important security aspect concerns the risk of correlation attacks on two or more vault records of the same subject. The correlation attack of [SB07, KY08] exploits that a correlation of the points from two vault records of the same biometric instance (fingerprint) will mostly result in genuine points. However, since we use all grid points unoccupied by minutiae as chaff points, all vault records contain the same set of quantized minutiae positions, namely, all grid points. This implies that a correlation between vault records on the basis of the quantized minutiae positions will always yield all vault points and, hence, gives no information. On the other hand, the attacker may try to perform correlation on the basis of the remainders (modulo α) of the minutiae angles stored in all vault points. However, for the optimal value $\alpha = 30$ (see 5), these remainders contain a lot of noise and only limited information, i.e., have a poor signal-to-noise-ratio. Nevertheless, we acknowledge that a correlation attack based on correlation of the angle's remainders must be taken into account and leave this aspect, as well as potential countermeasures, e.g. choosing the remainders for chaff points according to a more realistic distribution, to future investigations.

5 Experiments

In order to determine optimal parameters and the corresponding error rates, we performed experiments using the MCYT-100 database [OGFAS03]. Precisely, we used 1200 optical scans of the right index fingers captured with a Digital Persona UareU sensor from 100 subjects; these fingerprints have an image size of 256x400 pixels and a resolution of 500 dpi. Feature extraction was performed using the FingerJetFX minutiae extractor [FJF11]. For the determination of the False Non-Match Rate (FNMR), we enrolled every fingerprint and, for each vault record, performed verification attempts with all other fingerprints of the same finger, resulting in up to $100 \cdot 12 \cdot 11 = 13.200$ genuine comparisons. In order to estimate the FMR, we enrolled 6 fingerprints per subject and conducted 99 impostor verifications (with distinct subjects) per vault record, resulting in up to $100 \cdot 6 \cdot 99 = 59.400$ impostor comparisons.

We found the best trade-off (for variable k) between the FNMR and the security against recovery attacks for parameters $t = 35$, $q_x = 15$, $q_y = 20$ and $\alpha = 30$, which results in a vault size $r = 651$. For these parameters and varying k, Table 1 summarizes the observed error rates and the security (in bits) against the brute-force attack (SEC_{BF}), statistical attack (SEC_{ST}) and false-accept attack (SEC_{FA}), estimated as described in Section 4. The Failure-

to-enrol rate (FTE) was zero as no failures occurred in feature (minutiae) extraction and our implementation does not impose any quality requirements, e.g. a minimum number of minutiae, on the feature set.

Tab. 1: Error rates and security against relevant attacks of the scheme.

k	FNMR	FMR	SEC_{BF}	SEC_{ST}	SEC_{FA}
4	6.2%	1.56%	35.5	17.5	19.6
5	8.2%	0.16%	44.4	23.9	24.0
6	12.2%	0.02%	53.4	29.5	28.1
7	18.7%	0%	62.5	37.6	32.1

It turned out that the statistical attack is most efficient for security levels up to 24 bits, while, for higher security, the false-accept attack performs better. However, we stress that the efficiency of the statistical attack may be reduced by choosing the (angle's) remainders for chaff points according to a more realistic distribution. The effectiveness of that countermeasure and its impact to the error rates and security against false accept attacks remain to be investigated.

Tab. 2: Error rates and security against relevant attacks of the scheme without using minutiae types.

k	FNMR	FMR	SEC_{BF}	SEC_{ST}	SEC_{FA}
5	6.0%	1.9%	39.4	16.5	19.8
6	7.4%	0.3%	47.4	20.1	23.4
7	9.6%	0.04%	55.5	24.2	27.1
8	13.1%	0%	63.6	27.3	31.0

We also evaluated the error rates and security for the case that minutiae types are discarded (see Table 2). For this reduced implementation (and using accordingly adapted estimates for attacks), we observed a comparable recognition performance in terms of relation between FNMR and FMR and even a slightly better FMR for a given security level against the false-accept attack, which indicates that the recognition of types is too unreliable to improve the error rates. However, the statistical attack becomes more efficient and beats the false-accept attack for all security levels. Therefore, we conclude that, by discarding the minutiae types, the overall security, i.e., security against the most efficient attack, is slightly decreased for reasonably limited FNMR.

Finally, we compare our scheme with the fuzzy fingerprint vault of [Me11], which uses minutiae positions (without quantization), the multi-finger fuzzy vault of [TMM15], which uses minutiae positions quantized to a hexagonal grid, and the fuzzy vault presented in [Ta15b], which combines quantized minutiae positions (in polar coordinates) and angles (but no types) with three different local descriptors of the minutiae. Our scheme as well as those from [TMM15] and [Ta15b] rely on the directed reference point estimation method from [Ta13a] to address the alignment of minutiae positions and angles.

It is important to note that the scheme of [Me11] is vulnerable to the correlation and recovery attack described in [SB07, KY08], while the other schemes are resistant against this attack.

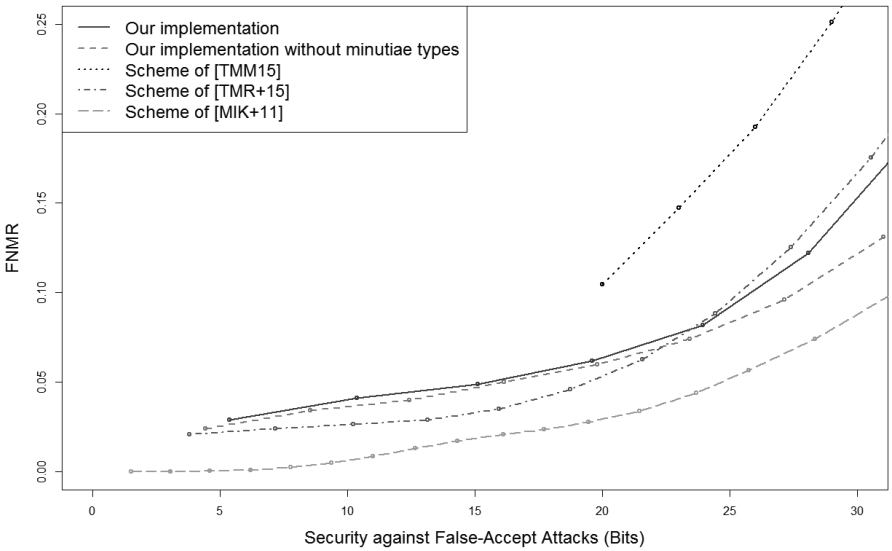


Fig. 3: Comparison of our fuzzy fingerprint vault implementation with those of [Me11], [TMM15] and [Ta15b] with respect to FNMR and security against false-accept attacks.

Figure 3 shows plots of the FNMR versus the security against false-accept attacks (depending on k) of the security and FNMR of our implementation, with and without using minutiae types, and for the schemes of [Me11], [TMM15] (in the single-finger setting) and [Ta15b]. For [Me11] and [Ta15b], we evaluated the error rates using the original implementations; in the latter scheme, we used the parameters reported as optimal in [Ta15b], while, for the former, we determined optimal parameters for an FTE below 5% in the single-finger setting.[7] Since the implementation of [TMM15] was not available to us, we used the results reported therein.

The comparison shows that our scheme achieves a much lower FNMR at the same security level than that of [TMM15]. Since both schemes apply the same pre-alignment and perform a quantization of the minutia positions, we attribute this improvement to the incorporation of minutia angles (and types), which were not used in [TMM15]; in fact, we observed error rates similar to that of [TMM15] when using only the positions of the minutiae.

When comparing our scheme to that of [Ta15b], our scheme has a higher FNMR for security levels under 23 bits, but performs considerably better for higher security levels, in particular, if minutiae types are not used. Both schemes have a significant higher FNMR

[7] In [Me11], two or more fingers were used.

than the implementation of [Me11]; we believe that the higher FNMR is a consequence of the quantization of the feature data, the use of all unoccupied points as chaff,[8] and the absolute pre-alignment. In particular, the reference point estimation of [Ta13a], which is used for absolute alignment in our scheme and in that of Tams et al., fails and gives highly inaccurate reference points for a small fraction of fingerprints, which explains the 'baseline' FNMR of $2\% - 2.5\%$ (even for $k = 1$) in both schemes. On the other hand, quantization and absolute pre-alignment are needed to achieve resistance against correlation attacks and, thus, we conclude that correlation-resistance comes at the price of reduced recognition accuracy.

6 Conclusion

In this paper, a step towards a correlation-resistant fuzzy vault is proposed that is based on the usage of all points of a quantization grid to achieve correlation-resistance. The fuzzy vault construction in this work also employs minutiae angles and types, in addition to minutiae positions. The results show that the use of more minutiae level information (positions, angles and types) have led to lower error rates and also higher security against brute force or false accept attacks. It turns out that minutiae types slightly deteriorate the results. The comparison of the results with previous works show the validity of the concept.

References

[CKL03] Clancy, T. Charles; Kiyavash, Negar; Lin, Dennis J.: Secure Smartcard-Based Fingerprint Authentication. In: Proc. ACM SIGMM workshop on Biometrics methods and applications. ACM, New York, NY, USA, pp. 45–52, 2003.

[CS09] Cavoukian, Ann; Stoianov, Alex: Biometrics: theory, methods, and applications. John Wiley & Sons, Inc., Hoboken, NJ, USA, chapter 26 - Biometric Encryption: The New Breed of Untraceable Biometrics, 2009.

[Do03] Dodis, Yevgeniy; Ostrovsky, Rafail; Reyzin, Leonid; Smith, Adam: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. IACR Cryptology ePrint Archive, 2003:235, 2003.

[FJF11] FingerJetFX OSE – Fingerprint Feature Extractor, Open Source Edition. README file, 2011.

[IS11] ISO/IEC 24745:2011: , Information Technology – Security Techniques – Biometric Information Protection. International Organization for Standardization, 2011.

[JS02] Juels, A.; Sudan, M.: A Fuzzy Vault Scheme. In: Proc. Int. Symp. Inf. Theory. p. 408, 2002.

[KY08] Kholmatov, A.; Yanikoglu, B.: Realization of Correlation Attack Against the Fuzzy Vault Scheme. In: Proc. SPIE. volume 6819, 2008.

[8] For the scheme of [Me11], we used 85 chaff points, and their vicinities (where query minutiae are considered to match the point) cover approximately 20% of the relevant fingerprint area.

[Li08] Li, Jianjie; Yang, Xin; Tian, Jie; Shi, Peng; Li, Peng: Topological structure-based alignment for fingerprint Fuzzy Vault. In: Proc. Int. Conf. on Pattern Recognition. pp. 1–4, 2008.

[Li10] Li, P.; Yang, X.; Cao, K.; Tao, X.; Wang, R.; Tian, J.: An alignment-free fingerprint cryptosystem based on fuzzy vault scheme. J. Netw. Comput. Appl., 33:207–220, 2010.

[Ma09] Maltoni, D.; Maio, D.; Jain, A.K.; Prabhakar, S.: Handbook of Fingerprint Recognition. Springer Publishing Company, Incorporated, 2nd edition, 2009.

[Me11] Merkle, J.; Ihmor, H.; Korte, U.; Niesing, M.; Schwaiger, M.: Performance of the Fuzzy Vault for Multiple Fingerprints. In: Proc. BIOSIG 2011. volume 191 of LNI. GI, pp. 57–72, 2011.

[MMT09] Mihăilescu, Preda; Munk, Axel; Tams, Benjamin: The Fuzzy Vault for Fingerprints is Vulnerable to Brute Force Attack. In: Proc. of BIOSIG 2009. volume 155 of LNI. GI, pp. 43–54, 2009.

[NJP07] Nandakumar, K.; Jain, A. K.; Pankanti, S.: Fingerprint-Based Fuzzy Vault: Implementation and Performance. IEEE Transactions on Information Forensics and Security, 2(4):744–757, 2007.

[NNJ10] Nagar, A.; Nandakumar, K.; Jain, A. K.: A hybrid biometric cryptosystem for securing fingerprint minutiae templates. Pattern Recogn. Lett., 31:733–741, June 2010.

[OGFAS03] Ortega-Garcia, J.; Fierrez-Aguilar, J.; Simon *et al.*, D.: MCYT baseline corpus: a bimodal biometric database. IEE Proc. on Vision, Image and Signal Processing, 150(6):395–401, 2003.

[SB07] Scheirer, W. J.; Boult, T. E.: Cracking Fuzzy Vaults and Biometric Encryption. In: Proc. of Biometrics Symp. pp. 1–6, 2007.

[Ta13a] Tams, Benjamin: Absolute Fingerprint Pre-Alignment in Minutiae-Based Cryptosystems. In: Proc. of BIOSIG 2013. volume 212 of LNI. GI, pp. 75–86, 2013.

[Ta13b] Tams, Benjamin: Attacks and Countermeasures in Fingerprint Based Biometric Cryptosystems. CoRR, abs/1304.7386, 2013.

[Ta15a] Tams, B.: Unlinkable Minutiae-Based Fuzzy Vault for Multiple Fingerprints. IET Biometrics, 2015. to appear.

[Ta15b] Tams, Benjamin; Merkle, Johannes; Rathgeb, Christian; Wagner, Johannes; Korte, Ulrike; Busch, Christoph: Improved Fuzzy Vault Scheme for Alignment-Free Fingerprint Features. In: Proc. of BIOSIG 2015. volume 245 of LNI. GI, 2015.

[TMM15] Tams, B.; Mihăilescu, P.; Munk, A.: Security Considerations in Minutiae-based Fuzzy Vaults. IEEE Trans. Inf. Forensics Security, 10(5):985–998, 2015.

[UJ06] Uludag, Umut; Jain, Anil K.: Securing fingerprint template: fuzzy vault with helper data. In: Proc. Workshop on Privacy Research In Vision. pp. 163–169, 2006.

[UPJ05] Uludag, Umut; Pankanti, Sharath; Jain, Anil K.: Fuzzy vault for fingerprints. In: Proc. Int. Conf. on Audio- and Video-Based Biometric Person Authentication. pp. 310–319, 2005.

[YV05] Yang, Shenglin; Verbaudwhede, Ingrid: Automatic secure fingerprint verification system based on fuzzy vault scheme. In: Proc. Int. Conf. on Acoustics, Speech and Signal Processing. pp. 609–612, 2005.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 137

Für bare Münze? NutzerInnenerfahrungen mit Sicherheit und Datenschutz bei Bitcoin[1]

Katharina Krombholz, Aljosha Judmayer, Matthias Gusenbauer, Edgar Weippl[2]

Abstract:

Im vorliegenden Paper wird die erste großangelegte NutzerInnenstudie zu Bitcoin vorgestellt. Dabei wird untersucht, wie die NutzerInnen das Bitcoin-Ökosystem im Hinblick auf Sicherheit, Datenschutz und Anonymität bewerten. 990 Bitcoin-NutzerInnen wurden mittels Online-Fragebogen befragt, um Strategien der Bitcoin-Verwaltung zu identifizieren und festzustellen, welche Sicherheitsmaßnahmen NutzerInnen zum Schutz ihrer Schlüssel und Bitcoins ergreifen. Dabei stellte sich heraus, dass 46 % der TeilnehmerInnen webbasierte Lösungen zur Verwaltung von Bitcoins verwenden und etwa die Hälfte davon ausschließlich solche Lösungen verwendet. Es zeigte sich auch, dass viele NutzerInnen die Sicherheitsmaßnahmen ihres Bitcoin-Verwaltungstools nicht voll ausschöpfen und falsch informiert sind, was die Anonymität und den Schutz ihrer Daten im Bitcoin-Netzwerk angeht. 22 % der TeilnehmerInnen haben bereits durch Sicherheitsvorfälle oder eigenes Verschulden einen finanziellen Verlust erlitten. Zum besseren Verständnis der beobachteten Phänomene wurden zusätzlich qualitative Interviews geführt. Außerdem wird eine umfassende Methode zur Kategorisierung von Bitcoin-Clients vorgestellt, mit der NutzerInnen rasch einen Überblick über die technischen Eigenschaften verschiedener Tools erhalten und leicht abschätzen können, was das für die Kontrolle und Verifizierbarkeit durch die NutzerIn bedeutet.

1 Einleitung

Mit einer Marktkapitalisierung von über 3,5 Milliarden US-Dollar ist Bitcoin die derzeit erfolgreichste Kryptowährung. Bitcoin wird für ca. 130.000 Transaktionen täglich verwendet [Bl14] und ist immer wieder in den Medien präsent. Nach dem Erfolg von Bitcoin sind mehrere weitere Kryptowährungen entstanden, sowohl basierend auf Bitcoin als auch Neuentwicklungen.

Obwohl Kryptowährungen zunehmend an Beliebtheit gewinnen, sind sie noch kein Massenphänomen. Einer der Gründe dafür ist, dass sich Bitcoin-NutzerInnen gezwungenermaßen mit kryptografischen Grundlagen und Schlüsselverwaltung auseinandersetzen müssen, wodurch die Verantwortung für den Großteil der Sicherheitsmaßnahmen im Gegensatz zu Zentralwährungssystemen auf den Endnutzer übergeht. Obwohl verschiedenste Software zur Verwaltung von Bitcoins verfügbar ist, müssen die NutzerInnen sich dennoch mit den technischen Grundlagen auseinandersetzen und Backups erstellen, um im Fall eines Verlustes ihr virtuelles Geld wiederherstellen zu können. Daher weisen diese Systeme

[1] Das Paper ist eine deutsche Version des Papers: Katharina Krombholz, Aljosha Judmayer, Matthias Gusenbauer, and Edgar Weippl. The Other Side of the Coin: User Experiences with Bitcoin Security and Privacy. In Proceedings of Financial Cryptography 2016, Barbados, February 2016.

[2] SBA Research, 1040 Wien, Österreich, (firstletterfirstname)(lastname)@sba-research.org

keine Resilienz im Bezug auf menschliches Versagen auf. Berichte aus Onlineforen und von Mailinglisten zeigen, dass viele Bitcoin-NutzerInnen bereits aufgrund der mangelnden Benutzerfreundlichkeit des Schlüsselmanagements sowie durch Sicherheitsvorfälle, etwa betrügerische Online-Börsen oder Wallets, Verluste erlitten haben. Daher ist es notwendig, das Interaktionsverhalten von NutzerInnen mit dem Bitcoin-Ökosystem zu untersuchen.

Bitcoin-NutzerInnen steht eine große Bandbreite an Werkzeugen zur Verwaltung ihres virtuellen Vermögens zur Verfügung. Diese werden üblicherweise als 'Wallets' bezeichnet. Ein Wallet wurde ursprünglich als Sammlung privater Schlüssel definiert [Es15]. Insofern könnte auch ein Stück Papier, auf dem der Schlüssel steht, oder sogar eine mentale Repräsentation desselben als Wallet bezeichnet werden. Die meisten dieser Tools verfügen aber neben der Schlüsselverwaltung auch über weitere Funktionalitäten, z.B. das Durchführen von Transaktionen. Im Gegensatz zu anderen kryptographischen Systemen, die auf öffentlichen Schlüsseln beruhen, z.B. PGP/GPG, ist Bitcoin nicht völlig kanalagnostisch. Die Interaktion mit dem Bitcoin-Netzwerk ist zwingend notwendig, um im verteilten System arbeiten zu können. Im Gegensatz zu anderen Signatursystemen müssen Bitcoin-Werkzeuge die Statusinformation durchgeführter Transaktionen bzw. Kontostände speichern.

Um Missverständnisse bei der Definition des Bitcoin-Wallets zu vermeiden, soll hier der weitreichendere Begriff *Coin Management Tool (CMT)* eingeführt werden. Ein CMT werde als Werkzeug oder Paket von Werkzeugen definiert, die es NutzerInnen erlauben, eine oder mehrere der Kernaufgaben von Kryptowährungen zu verwalten. In diesem Artikel soll daher der Begriff *Bitcoin-Verwaltung* verwendet werden, da er eine bessere Beschreibung der Aktivitäten von NutzerInnen bei der Interaktion mit dem Bitcoin-Ökosystem darstellt. Sicherheits- und Datenschutzaspekte von Bitcoin wurden bereits in der Vergangenheit untersucht [Bo15, GKL15, He15, Ge, Go]. Eine erste Untersuchung der Schlüsselverwaltung bei Bitcoin wurde in [Es15] vorgestellt. Der vorliegende Artikel beschreibt jedoch die erste umfassende NutzerInnenstudie, bei der Erfahrungen von NutzerInnen mit der Sicherheit und dem Datenschutz von Bitcoin erhoben werden.

Es handelt sich dabei um eine umfassende NutzerInnenstudie ($n = 990$) zur Interaktion von Mensch und Computer im Bitcoin-Ökosystem. Ziel der Untersuchung war es, zu verstehen, wie NutzerInnen mit Bitcoin interagieren und wie sie ihr virtuelles Vermögen verwalten. Außerdem wurden Erfahrungen und Wahrnehmungen zu Sicherheit, Datenschutz und Anonymität im Bitcoin-Netzwerk gesammelt. Die Datenerhebung wurde mittels einer Online-Umfrage mit 990 TeilnehmerInnen durchgeführt. Zusätzlich wurden mit 10 davon qualitative Interviews durchgeführt.

Im Zuge der Studie wurden interessante Erkenntnisse zur Interaktion von NutzerInnen mit dem Bitcoin-Netzwerk sowie zu den Datenschutz- und Sicherheitsmechanismen, die sie zum Schutz ihrer Schlüssel und ihres Vermögens nutzen, gewonnen. Dabei zeigte sich, dass sich an Platz 1 und 3 der meistverwendeten CMTs mit Coinbase und Xapo webbasierte Tools befinden, bei denen die NutzerInnen die Verantwortung für die Sicherheit an eine dritte Partei abgeben. Es stellte sich auch heraus, dass etwa ein Drittel der NutzerInnen dieser Tools nicht wusste, ob ihre CMT-Daten verschlüsselt und Sicherungskopien erstellt werden. Von den TeilnehmerInnen, die eine webbasierte Lösung verwendeten, ga-

ben 50 % an, dass sie ausschließlich dieses Tool verwendeten, während die andere Hälfte zusätzlich lokale Clients zur Verwaltung ihrer Bitcoins einsetzte. In Bezug auf Risikoszenarien und deren Eintrittswahrscheinlichkeit wurde das zweithöchste Risiko Schwachstellen in webbasierten CMTs zugeschrieben (nach Wertschwankungen und vor Diebstahl durch Malware).

Es zeigte sich auch, dass viele NutzerInnen eine falsche Vorstellung davon haben, wie sie anonym bleiben können. Ca. 25 % der TeilnehmerInnen gaben an, auf Bitcoin über das Tor-Netzwerk zuzugreifen, was sich in bestimmten Fällen ein Sicherheitsproblem darstellen kann [BP14, Al14]. 22,5 % der TeilnehmerInnen gaben an, Bitcoins durch Sicherheitsvorfälle verloren zu haben. Etwa die Hälfte davon sehen diesen Verlust als ihre eigene Schuld an und die meisten von ihnen konnten ihre Bitcoins nicht wiederherstellen und machten so einen permanenten Verlust. Die vorliegende Studie ist ein Beitrag zum Wissen über nutzerInnenzentrierte Bedenken beim Bitcoin-Management, da laut Bonneau [Bo15] im Falle von Bitcoin die Praxis der Theorie voraus ist.

2 Stand der Forschung

Unsere Studie baut auf früheren Forschungsarbeiten auf. Eskandari et al. [Es15] präsentierten einen ersten Blick auf die Schlüsselverwaltung bei Bitcoin durch eine Reihe von Evaluierungskriterien für Bitcoin-Wallets. Außerdem werden Usability-Aspekte der Software von Bitcoin-Wallets in einem Cognitive Walkthrough [Wh94] evaluiert. Die Studie von Eskandari et al. [Es15] kann als erste Untersuchung der Usability von Bitcoin angesehen werden. Bisher wurde keine empirische Studie mit Bitcoin-NutzerInnen zu Aspekten wie Sicherheit und Nutzbarkeit vorgestellt. Für Kryptowährungen wie Bitcoin ist eine Chiffrierung mit öffentlichem Schlüssel notwendig. Im Bereich von E-Mail haben hier zahlreiche Studien gezeigt, dass es bei der Usability von Schlüsselverwaltung und Verschlüsselung noch viele Probleme gibt [WT99, GM05, Ga05, Sh06]. Bislang gibt es für keinen dieser zwei Bereiche ein komplett anwendbares erfolgreiches Konzept. Die menschlichen Faktoren bei der Schlüsselverwaltung wurden bereits in anderen Bereichen untersucht [WT99, Ga05, GM05, Sh06, GFFK06]). Im Bitcoin-Ökosystem ist allerdings die sichere Schlüsselverwaltung allein nicht ausreichend, da die Kommunikation nicht kanalunabhängig ist, sondern einen elementaren Bestandteil des Sicherheitskonzepts darstellt.

3 NutzerInnenstudie: Methodologie

Ziel der Studie ist eine empirische Untersuchung der Sichtweise von EndnutzerInnen und deren Verhalten im Bitcoin-Ökosystem mit einem Schwerpunkt auf Sicherheitsmaßnahmen sowie die Verwaltung von Bitcoins und Schlüsseln mit den damit verbundenen Sicherheitsrisiken. Dazu wurde ein Online-Fragebogen erstellt, der durch qualitative Interviews ergänzt wurde. Die Forschungsfragen wurden auf Grundlage der bestehenden Literatur zu Bitcoin (s. 2) sowie einer qualitativen Inhaltsanalyse von Threads in Onlineforen und Mailinglisten formuliert. Außerdem wurden die verfügbaren Bitcoin-Wallets

und deren Leistungsumfang untersucht und als Inspiration für die Fragen sowie zur Entwicklung der Risikoszenarien verwendet. Der Fokus der Studie liegt auf Bitcoin, da dies zum Durchführungszeitpunkt (Juli 2015) die mit Abstand beliebteste Kryptowährung war. Während der Online-Fragebogen die Angaben der NutzerInnen zu Bitcoin-Verwaltungsverhalten und Risikowahrnehmung erheben sollte, wurden die zusätzlichen Interviews mit Bitcoin-NutzerInnen zum tiefergehenden Verständnis der wichtigsten Usability-Probleme und der Gründe für übliche Sicherheitsvorfälle geführt sowie auch, um zu eruieren, ob und wie verlorene Schlüssel wiederhergestellt werden konnten.

Mit der Studie sollten die folgenden Forschungsfragen zur NutzerInnenwahrnehmung in den Themenbereichen Verwaltung von Bitcoins und mit Bitcoins verbundene Sicherheitsrisiken beantwortet werden:

- *Q1: Was sind die wichtigsten Verwendungsszenarien für Bitcoins?*
- *Q2: Wie verwalten die TeilnehmerInnen ihre Bitcoins? Wie verhalten sich TeilnehmerInnen im Hinblick auf Sicherheit, Datenschutz und Anonymität?*
- *Q3: Wie nehmen die TeilnehmerInnen mit Bitcoin verbundene Sicherheitsrisiken wahr?*
- *Q4: Von welchen Sicherheitsvorfällen waren NutzerInnen in der Vergangenheit betroffen und wie bekamen sie ihre Bitcoin-Schlüssel und Bitcoins wieder?*
- *Q5: Was sind die größten Herausforderungen an Usability, denen sich NutzerInnen bei der Verwendung von Bitcoin stellen müssen?*

Detaillierte Information zu den Online-Umfragen,den Qualitativen Interviews sowie die genauen Fragestellungen des Fragebogens finden sich im englischen Originalpapier.

4 Ergebnisse

In diesem Abschnitt werden die erhobenen Antworten analysiert, um die in Abschnitt 3. definierten Forschungsfragen zu beantworten. Zu Beginn jedes Abschnittes werden zunächst die Ergebnisse der Online-Umfrage analysiert und dann den Ergebnissen aus den qualitativen Interviews gegenübergestellt, um die Erkenntnisse miteinander in Bezug zu setzen und zu erklären.

4.1 Allgemeine Verwendung von Bitcoin (Q1)

Die meisten TeilnehmerInnen gaben an, Bitcoin für Spenden zu verwenden (38,0 %), gefolgt von virtuellen Gütern und Dienstleistungen wie Webhosting oder Online-Zeitungen (33,3 %) Online-Shopping (27,5 %), alternative Kryptowährungen (altcoins) (26,5 %), Glücksspiel (26,5 %) und Bitcoin-Geschenkgutscheine (19,9 %). Etwa 5 % gaben an, mit Bitcoins Drogen zu kaufen oder dies in der Vergangenheit getan zu haben. 30,2 % gaben an, Bitcoin mindestens einmal pro Woche zu verwenden, 25 % verwenden Bitcoin mindestens einmal pro Monat und 19 % mindestens einmal täglich. Die übrigen TeilnehmerInnen gaben an, Bitcoin mindestens einmal pro Jahr oder noch seltener zu verwenden.

Die Ergebnisse lassen den Schluss zu, dass die Mehrheit der StudienteilnehmerInnen Bitcoin häufig verwendet. Die TeilnehmerInnen wurden auch nach ihrem aktuellen Bitcoin-Kontostand gefragt. Etwa die Hälfte der TeilnehmerInnen wollte hier keine Angabe machen. Die TeilnehmerInnen, die diese Frage beantworteten, haben ein Gesamtvermögen von etwa 8000BTC. Die Mehrzahl der TeilnehmerInnen (70 %) begann zwischen 2013 und 2015 mit der Verwendung von Bitcoin. 17 % begannen zwischen 2011 und 2012. 58,0 % gaben an, neben Bitcoin auch andere Kryptowährungen zu nutzen, allen voran Dogecoin und Litecoin. Die beliebtesten Bitcoin-Börsen waren BTCE (20,9 %), Bittrex (14,0 %) und Bitstamp (13,0 %). 11,4 % der TeilnehmerInnen sind derzeit am Mining von Bitcoins beteiligt. Die meisten begannen damit nach 2014. Vielen von denen, die früher mit dem Mining begannen, haben damit inzwischen wieder aufgehört, da sie der Meinung sind, dass es sich derzeit nicht lohnt. 195 TeilnehmerInnen (19,7 %) gaben an, einen vollen Bitcoin-Server zu betreiben, der über das Internet erreichbar ist. Als Hauptgrund dafür wurde angegeben, das Bitcoin-Netzwerk unterstützen zu wollen (60,5 %), gefolgt von der schnellen Verbreitung von Transaktionen (46,6 %), Netzwerkanalyse (30,3 %) und dem Erkennen von Double Spend-Attacken (26,1 %).

In den qualitativen Interviews gaben alle Teilnehmer an, Bitcoin häufig zu verwenden. Einige von ihnen sind auch im lokalen Bitcoin-Verein aktiv. Die meisten Interviewpartner nannten den dezentralen Charakter von Bitcoin als einen der Hauptgründe dafür, warum sie mit der Verwendung von Bitcoin begonnen hatten. Der zweithäufigste Grund war Neugier. Ein Teilnehmer, der zur Zeit des Ausbruchs des Konflikts zwischen der Ukraine und Russland auf der Krim lebte, nannte soziopolitische Gründe. Er arbeitete dort damals für ein US-Unternehmen und benötigte eine sichere und günstige Option, um sein Gehalt zu erhalten. Außerdem wollte er sichergehen, dass er durch die Annexion der Krim durch die Russische Föderation kein Geld verlieren würde. Für ihn war Bitcoin die beste Option und er sagte, dass zu der Zeit auf der Krim viele Menschen begonnen hätten, Bitcoin zu verwenden. Einige der TeilnehmerInnen waren vor einigen Jahren am Bitcoin-Mining beteiligt, als das Mining im kleinen Maßstab noch gewinnbringend war.

4.2 Bitcoin-Verwaltung (Q2)

4.2.1 Bitcoin-Wallets und Backup-Verhalten.

Tabelle 2 zeigt die am häufigsten verwendeten Bitcoin-Wallets. Bei dieser Frage waren Mehrfachantworten möglich, da das Verwenden von mehr als einem Wallet üblich ist. Die Tabelle zeigt die Anzahl sowie den Prozentsatz der TeilnehmerInnen, die ein bestimmtes Wallet verwenden. Außerdem zeigt Tabelle 2 sie, ob die NutzerInnen ihre Wallets mit einem Passwort schützen und ob sie verschlüsselt sind. Die Studie zeigte, dass die Mehrheit der NutzerInnen ihre Wallets mit einem Passwort schützen. Im Fall von Webclients wurde fehlendes Hintergrundwissen festgestellt. So gaben 47,7 % der Coinbase-NutzerInnen in der Umfrage an, ihr Wallet sei verschlüsselt, während 34 % sagten, sie wüssten nicht, ob es verschlüsselt sei. Bei Xapo, dem drittbeliebtesten Wallet in dieser Stichprobe, ist ein ähnlicher Trend festzustellen. Wie Coinbase ist es ein webbasiertes

Tool und ähnlich wie bei Coinbase gibt nur etwa die Hälfte der NutzerInnen an, dass es verschlüsselt sei, während ein Drittel es nicht weiß. Nur ein Drittel der Coinbase-NutzerInnen und 43 % der Xapo-NutzerInnen erstellen Sicherheitskopien von ihren Wallets. 33,9 % der Coinbase-NutzerInnen und 28,5 % der Xapo-NutzerInnen wissen nicht, ob von ihrem Wallet ein Backup besteht. Es stellte sich auch heraus, dass Bitcoin-NutzerInnen mit mehr als 0.42BTC (100 USD) von ihrem CMT nicht häufiger ein Backup erstellen als NutzerInnen mit weniger Bitcoins. Der Effekt ist statistisch signifikant ($\chi^2(1) = 5.1, p = 0.02$).

Die TeilnehmerInnen wurden auch gefragt, ob sie für den Fall eines Verlustes oder Diebstahls des primären Backups zusätzliche Backups erstellen. Bitcoin Core-NutzerInnen gaben dabei am häufigsten an, zusätzliche Backups zu erstellen - bei 64,0 % von ihnen war dies der Fall. Tabelle 1 zeigt die Angaben der NutzerInnen zu den Eigenschaften der Wallet-Backups. Den Ergebnissen der Studie zufolge speichert niemand das Backup auf einem durch Air Gap geschützten Computer. Die häufigsten genannten Eigenschaften der Backups waren Verschlüsselung und Passwortschutz. 197 Backups waren in einer Cloud gespeichert.

59,7 % der TeilnehmerInnen verwenden nur ein einziges Wallet zur Verwaltung ihrer Bitcoins, 22,7 % verwenden zwei und 10,6 % drei Wallets. Die übrigen 7 % verwenden vier oder mehr Wallets. Die höchste angegebene Zahl an Wallets war 14, wobei dieser Teilnehmer die Erklärung hinzufügte, er habe zuerst die verschiedenen Wallets ausprobieren wollen, um dann jene auszuwählen, die am besten seinen Bedürfnissen entsprachen. Etwa die Hälfte jener TeilnehmerInnen, die einen Web-Client benutzten, benutzten ausschließlich diesen zur Verwaltung ihrer Bitcoins, die andere Hälfte verwendete ihn zusätzlich zu einem lokalen Client.

Überraschenderweise zeigte sich, dass die Bitcoins der meisten TeilnehmerInnen in Armory gespeichert sind. Insgesamt hatten die Armory-NutzerInnen etwa 3818 BTC in ihren Armorys, wobei die fünf größten NutzerInnen angeben, 2000, 885, 300, 230 und 150 BTC zu haben. Die größte Summe an Bitcoins, die im Web-Client eines Teilnehmers gespeichert war, war 100 BTC. Insgesamt sind in Coinbase 238 BTC und in Xapo 157 BTC gespeichert.

Tab. 1: Eigenschaften der Backups (absolute Nennungen in absteigender Reihenfolge). Eine NutzerIn kann über mehrere Wallets und mehrere Backups verfügen.

Backup-Eigenschaften	**Nennungen**
Mein Backup ist verschlüsselt	662
Mein Backup ist passwortgeschützt	629
Mein Backup ist auf einem externen Speichermedium gespeichert (z.B. USB-Stick)	430
Mein Backup ist auf Papier gespeichert	334
Mein Backup ist in der Cloud gespeichert (z.B. Dropbox)	197
Mein Backup ist auf einem Air Gap-Gerät gespeichert	0

Tab. 2: Am häufigsten erwähnte CMTs nach Anzahl sowie Anteil der TeilnehmerInnen mit NutzerInnenangaben dazu, ob das CMT verschlüsselt ist, ob ein Backup davon erstellt wird und ob es ein zusätzliches Backup gibt (Ja (J), Nein (N), Weiß nicht (W.N.)) in Prozent der NutzerInnen. Die rechte Spalte zeigt die Gesamtsumme der Bitcoins, die die TeilnehmerInnen auf dem jeweiligen CMT gespeichert haben.

			Verschlüsselt?			Backup?			Zusätzliches Backup?			
CMT	**Anzahl**	**Prozent**	**J**	**N**	**W.N.**	**J**	**N**	**W.N.**	**J**	**N**	**W.N.**	**BTC**
Coinbase	314	31,7	47,5	18,5	34	35,5	30,6	33,9	30,3	66,9	2,8	238
Bitcoin Core	236	23,8	72,8	16,1	11,1	76,3	14,0	9,7	64,0	32,2	3,8	752
Xapo	179	18,1	51,4	19,0	29,9	43,0	28,5	28,5	41,3	57,5	1,2	157
Electrum	125	12,6	72,8	15,2	22,0	77,6	16	6,4	55,2	44	0,8	226
MyCelium	97	9,8	61,9	21,6	16,5	83,5	12,4	4,1	52,6	47,2	0,2	62

4.2.2 Anonymität.

Die Umfrage zeigte, dass 32,3 % der TeilnehmerInnen der Meinung sind, Bitcoin sei völlig anonym. Weitere 47,0 % sind der Meinung, dass Bitcoin per se zwar nicht anonym sei, aber anonym verwendet werden könnte. Allerdings sind etwa 80 % der Meinung, dass ihre Transaktionen nachverfolgt werden können. 25 % gaben an, Bitcoin zum Schutz ihrer Anonymität über Tor verwendet zu haben.

Die Frage, ob sie sich weiterer Anonymisierungsmaßnahmen bedienten, beantworteten 18 % mit “häufig”. Die meisten von ihnen gaben an, Bitcoin über Tor zu verwenden, gefolgt von mehreren Adressen, Mixing-Services, mehreren Wallets und VPN-Diensten. In der Forschung wurde bereits gezeigt, dass die Verwendung von Bitcoin über Tor einen Angriffsvektor erzeugt, der für deterministische Man-in-the-Middle Angriffe und Fingerprinting ausgenutzt werden kann. [BP14, Al14].

4.3 Risikowahrnehmung (Q3)

Ein weiteres Thema der Umfrage war die Wahrnehmung der Nutzerinnen zu Risiken von Bitcoin. Den TeilnehmerInnen wurden 11 Risikoszenarien vorgestellt, die auf Erkenntnissen aus der Literatur und Informationen aus Onlinequellen basierten. Jedes Risiko wurde leicht verständlich erklärt und die TeilnehmerInnen wurden gefragt, ob sie einen Risikoeintritt für wahrscheinlich oder unwahrscheinlich hielten. Abb. 1 zeigt die Einschätzung der TeilnehmerInnen zu den Risikoszenarien. Dabei zeigte sich, dass TeilnehmerInnen Kursschwankungen als höchstes Risiko einstufen, gefolgt von Schwachstellen in webbasierten Wallets und dem Diebstahl von Bitcoins mittels Malware. Das Risiko von Schwachstellen in der Kryptographie wurde als niedrigstes eingeschätzt, an zweitletzter Stelle standen Double Spending-Angriffe und davor DoS-Angriffe auf das Bitcoin-Netzwerk.

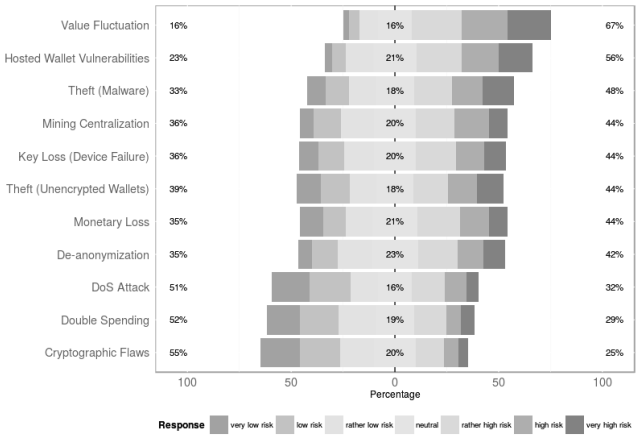


Abb. 1: NutzerInnenwahrnehmung der Risikowahrscheinlichkeit in Prozent der TeilnehmerInnen ($N = 990$).

4.4 Sicherheitsvorfälle (Q4)

Etwa 22,5 % gaben an, mindestens einmal einen Verlust von Bitcoins oder Bitcoin-Schlüsseln erlitten zu haben. Davon gaben 43,2 % als Grund einen selbstverschuldeten Fehler an, wie beispielsweise den Verlust des Geräts oder das versehentliche Löschen. Weitere genannte Risiken wurden wie folgt genannt: 26,5 % Hardware-Versagen, 24,4 % Software-Versagen. 18,0 % gaben an, ihre Schlüssel im Zuge eines größeren Sicherheitsvorfall durch Schadsoftware oder Hacker verloren zu haben. Die meisten von ihnen (77,6 %) wollten keine Angaben dazu machen, ob sie die Schlüssel wiederherstellen konnten. Von denen, die die Frage beantworteten, gelang es 65 % nicht, ihre Schlüssel wiederherzustellen. Insgesamt gaben die Teilnehmer an, etwa 660,6873 Bitcoins verloren zu haben. Dabei ist aber zu erwähnen, dass nicht nach dem Zeitpunkt des Verlustes gefragt wurde. Bei der Interpretation dieses Ergebnisses muss daher beachtet werden, dass der Bitcoin-Kurs starken Schwankungen ausgesetzt ist und es daher schwer ist, den Gesamtverlust in US-Dollar zu beziffern. Etwa 40 % der TeilnehmerInnen gaben an, Geld durch einen von ihnen als ernst eingestuften Sicherheitsvorfall verloren zu haben. 13,1 % der Gesamtstichprobe gaben an, durch HYIPS (high-yield investment programs) und Schneeballsysteme Bitcoins verloren zu haben. 7,9 % verloren auf Mt. Gox Bitcoins.

Die TeilnehmerInnen hatten auch die Möglichkeit, zu beschreiben, wie sie auf den Vorfall reagierten. Die meisten gaben an, nichts getan und den Verlust einfach akzeptiert zu haben. Einige sagten, der finanzielle Verlust sei so gering gewesen, dass es sich nicht gelohnt hätte, etwas zu unternehmen, oder dass sie sich hilflos fühlten und nicht wussten, was sie tun könnten. Jene, die etwas unternahmen, gaben meist an, Forderungen geltend gemacht und den Provider der Börse oder des Online-Wallets kontaktiert zu haben. Jene, die Geld an bösartige Online-Wallets verloren haben, gaben an, auf andere Wallet-Typen umgestiegen zu sein und keine webbasierten Wallets mehr zu verwenden. Die TeilnehmerInnen, die bei HYIPS Verluste machten, gaben meist an, dass sie aus ihren Fehlern gelernt hätten

und nun weniger risikoreiche Investitionen machten. Unabhängig von der Art des Sicherheitsvorfalls gaben viele TeilnehmerInnen an, darüber online in Foren gepostet und sich mit anderen Betroffenen ausgetauscht zu haben.

"Ich folge dem Motto 'investier nicht mehr, als du bereit bist, zu verlieren'." (P3848)

"Ich musste einfach akzeptieren, dass mein Geld gestohlen worden war ... und ich lernte daraus, Börsen niemals als Wallets zu verwenden. Man sollte alles bei sich behalten." (P3763)

"Ich habe einfach daraus gelernt. Es war unglaublich dumm von mir." (P853)

Bei den qualitativen Interviews gaben acht Personen an, bereits einmal durch einen Angriff oder einen Fehler Schlüssel und/oder Bitcoins verloren zu haben. Drei Teilnehmer waren vom Angriff auf Mt. Gox betroffen und zwei gaben an, gegen Kraken[5] Forderungen geltend gemacht zu haben. Ein Teilnehmer gab an, eine *physische* Casascius-Bitcoin-Münze[6] verloren zu haben; er habe aber aufgehört, danach zu suchen, da sie damals nur 9 US-Dollar wert gewesen sei. Andere gaben auch an, durch Geräteversagen, korrupte HDDs oder Softwareversagen Schlüsselverluste erlitten zu haben.

4.5 Bewertung der Usability (Q5)

Obwohl bei den qualitativen Interviews die meisten angaben, bei der Bitcoin-Verwaltung sehr auf Sicherheit und Datenschutz zu achten, sagten acht von ihnen, dass sie Bitcoin-AnwenderInnen ohne technisches Wissen zu webbasierten und deterministischen Wallets raten würden. Als Hauptgrund dafür wurde die einfache und praktische Handhabung genannt. Ein Teilnehmer sagte, er würde auf jeden Fall ein Wallet empfehlen, bei dem der private Schlüssel auf einem zentralen Server gespeichert ist, um den Schlüssel im Fall eines Verlustes leichter wiederherstellen zu können und so die Notwendigkeit von umfassenden Backups zu vermeiden und dass Gedächtnishilfen hilfreich wären. Sechs Teilnehmer sagten auch, dass sie MyCelium[7] als das am leichtesten verwendbare Wallet empfehlen würden. Jene, die bereits Erfahrung mit MyCelium hatten, waren der Meinung, der Papier-Backup-Prozess sei der sicherste und am nutzerfreundlichste dieser Art. Um mit MyCelium ein Papier-Backup zu machen, muss die NutzerIn ein Template ausdrucken, auf dem Teile des Schlüssels aufgedruckt sind, der dann von der NutzerIn händisch ergänzt werden muss. Einige Teilnehmer empfanden es anfangs als sehr ungewohnt, Papier-Wallets zu verwenden.

"Es fühlte sich irgendwie nicht ganz richtig an, den digitalen Raum zu verlassen." (P6 über Papier-Wallets)

Die meisten Teilnehmer strichen bei den Interviews die Notwendigkeit hervor, bereits in der Kindheit damit zu beginnen, das System zu erlernen. P2 sagte, Bitcoin sei inhärent

[5] `https://www.kraken.com/`
[6] `https://www.casascius.com/`
[7] `https://mycelium.com/`

komplex und dass die Grundidee von Verschlüsselung mit öffentlichen Schlüsseln in Schulen vermittelt werden sollte, und dass Währungssysteme Teil der Kultur seien.

"Kinder lernen schon in der Volksschule, wie unser Geldsystem funktioniert. Deswegen können wir als Gesellschaft Bargeld und Kreditkarten verwenden. Ich bin mir sicher, das könnte auch bei einer dezentralen Kryptowährung so sein." (P7)

Zwei Teilnehmer sagten außerdem, dass Anwenderoberflächen einfacher und minimalistisch sein sollten. Viele Teilnehmer sagten, dass für eine rasche Verbreitung von Bitcoin einfache und intuitiv zu bedienende Nutzeroberflächen wichtiger seien als Sicherheitsaspekte. Dies wurde damit begründet, dass auch Computer sich rasch verbreitet haben, obwohl die meisten Menschen nicht wissen, wie sie funktionieren, und dass Sicherheit bei der großflächigen Verbreitung nicht unbedingt ein Argument sei. Als Beispiele wurden Autos in den 1940er-Jahren, Computer, Kreditkarten und WhatsApp genannt. Sie sagten auch, dass die Summen, die im Bitcoin-Netzwerk zirkulieren, niedrig genug seien um das Risiko eines Verlustes eingehen zu können. Dieses Szenario wurde mit dem Verlustrisiko von Bargeld verglichen. Manche Teilnehmer schlugen auch vor, dass ein dediziertes Gerät für das Schlüsselmanagement mit einer intuitiven Nutzeroberfläche entwickelt werden könnte und waren der Meinung, dass so etwas die sicherste und nutzerfreundlichste Option wäre.

5 Diskussion

Unser Ziel war es, die in gestellten Forschungsfragen (3) zu beantworten, um zu verstehen, wie NutzerInnen mit Bitcoin interagieren. Bei dieser ersten NutzerInnenstudie mit Bitcoin-NutzerInnen wurden wertvolle Erkenntnisse gewonnen. Die Ergebnisse sollen hier in Beziehung zu bestehenden Forschungsarbeiten gestellt werden.

Bezüglich der Bitcoin-Verwaltung (Q2) zeigte sich, dass zwei der beliebtesten CMTs webbasierte Lösungen sind, bei denen die NutzerInnen sich nicht um Schlüsselverwaltung und Backups kümmern müssen. Interessanterweise wusste bei beiden Clients jeweils etwa ein Drittel nicht, ob ihr Wallet verschlüsselt ist oder ob ein Backup besteht. In diesem Szenario übertragen die NutzerInnen die Verantwortung an eine dritte Partei. Obwohl dies für nicht fachkundige NutzerInnen eine praktische und benutzerfreundliche Lösung zu sein scheint, bedeutet es, dass die NutzerIn darauf vertrauen muss, dass die dritte Partei sich um ihre Sicherheit kümmert. Etwa 50 % der Web-Client-NutzerInnen gaben an, zusätzlich einen lokalen Client zu verwenden, um ihr virtuelles Vermögen zu speichern. Die Antwort auf Q4 zeigt, dass ein bedeutender Teil der TeilnehmerInnen bereits Geld an bösartige Wallet-Provider verloren hat. Auch wurden Schwachstellen in webbasierten Wallets unter den Risikoszenarien an zweiter Stelle gereiht (Q5). Bei den qualitativen Interviews sagten einige Teilnehmer, dass sie unerfahrenen NutzerInnen empfehlen würden, zum Einstieg ein webbasiertes Wallet zu verwenden, da dies die nutzerfreundlichste Lösung sei. Bei den meisten anderen Lösungen müssen NutzerInnen zumindest die Grundlagen von Bitcoin und der Block-Chain verstehen.

Bitcoin ist ein pseudonymes System, in den Medien wird jedoch oft der Mythos verbreitet, es sei an sich anonym. Über ein Drittel der StudienteilnehmerInnen glaubt dies und gab an, zu denken, dass Bitcoin völlig anonym sei. Etwa die Hälfte der TeilnehmerInnen ist sich dessen bewusst, dass Bitcoin an sich nicht anonym ist, dass es aber anonym verwendet werden kann. Viele NutzerInnen gaben an, zur Wahrung ihrer Anonymität Bitcoin über Tor zu verwenden, wodurch aber ein Angriffsvektor für deterministische Man-in-the-Middle-Angriffe entsteht, wie in [BP14] gezeigt wurde. Die Ergebnisse deuten darauf hin, dass die StudienteilnehmerInnen auf die Kryptographie hinter Bitcoin vertrauen und sich der Risiken, die durch Kursschwankungen und Softwaresicherheitslücken entstehen können, bewusst sind. Schlechte Anwendbarkeit und mangelndes Wissen tragen stark zu Sicherheitsproblemen bei. Beinahe ein Viertel der TeilnehmerInnen gab an, bereits zumindest einmal einen Verlust von Bitcoins oder Schlüsseln erlitten zu haben (Q5). Überraschenderweise war beinahe die Hälfte dieser Verluste auf selbstverursachte Fehler zurückzuführen, was zeigt, dass moderne CMTs immer noch bisweilen schwer zu verwenden sind oder erfordern, dass NutzerInnen Sicherheitsmaßnahmen wie Backups und Verschlüsselung manuell durchführen. Die Studie zeigt, dass das Bitcoin-Ökosystem vor allem für Spenden sowie zum Erwerb digitaler Güter verwendet wird, zum Teil aber auch für kriminelle Aktivitäten und Glücksspiel.

6 Schlussfolgerungen

Im diesem Artikel wurde die erste Studie zu NutzerInnen-Interaktion mit dem Bitcoin-Ökosystem vorgestellt. Dabei wurde eine Online-Umfrage mit 990 NutzerInnen durchgeführt, die durch qualitative Interviews mit 10 Personen ergänzt wurden. Der Begriff *Coin Management Tool (CMT)* wurde eingeführt, um Tools zu beschreiben, mit denen NutzerInnen ihr virtuelles Vermögen verwalten und mit Bitcoin interagieren können. Es zeigte sich, dass die Verwaltung von Bitcoins weiterhin eine große Herausforderung darstellt, da viele von ihnen keine ausreichenden Sicherheitsmaßnahmen, wie Verschlüsselung oder Backups, durchführen. Viele der TeilnehmerInnen wussten nicht, welche Sicherheitsfeatures ihr CMT hatte. Zwei der beliebtesten CMTs unter den TeilnehmerInnen waren webbasierte Lösungen. Etwa die Hälfte der NutzerInnen dieser Lösungen gab an, nur Online-Clients zu verwenden, während die andere Hälfte sie mit lokalen Clients kombiniert. Obwohl Webclients eine nutzerfreundliche Lösung sein sollten, erfordern sie ein gewisses Maß an Vertrauen und übertragen die Verantwortung für Verschlüsselung und Backups an eine dritte Partei. Die Studie zeigte auch, dass 22,5 % der TeilnehmerInnen bereits Bitcoins verloren haben. Etwa die Hälfte davon gab als Grund eigenes Verschulden an, woraus ersichtlich ist, dass NutzerInnen es immer noch schwierig finden, ihre Bitcoins sicher zu verwalten. Wir sind davon überzeugt, dass unsere Erkenntnisse ein wichtiger erster Schritt zur Verbesserung der Sicherheit und Nutzbarkeit von Bitcoin sind. Um sichere Interaktionen mit dem Bitcoin-Ökosystem zu garantieren, müssen NutzerInnen, sowohl jene mit als auch jene ohne Fachwissen, das Konzept der Bitcoin-Verwaltung neu überdenken, da es um weit mehr geht als die Sicherung der geheimen Schlüssel. Bitcoin ist ein dezentrales System in dem die Interaktion zwischen Peers und die Verbreitung und Verifizierung von Nachrichten und Daten wichtig sind. Würde dieser Aspekt ignoriert, wären Bitcoin lediglich signierte Zahlen ohne Wert.

Literaturverzeichnis

[Al14] Alex Biryukov and Dmitry Khovratovich and Ivan Pustogarov: Deanonymisation of clients in Bitcoin P2P network. CoRR, abs/1405.7418, 2014.

[Bl14] Blockchain.info: , Bitcoin currency statistics, April 2014. Accessed: 2014-04-05.

[Bo15] Bonneau, Joseph; Miller, Andrew; Clark, Jeremy; Narayanan, Arvind; Kroll, Joshua A; Felten, Edward W: SoK: Research Perspectives and Challenges for Bitcoin and Cryptocurrencies. 2015.

[BP14] Biryukov, Alex; Pustogarov, Ivan: Bitcoin over Tor isn't a good idea. arXiv preprint arXiv:1410.6079, 2014.

[Es15] Eskandari, Shayan; Barrera, David; Stobert, Elizabeth; Clark, Jeremy: A first look at the usability of bitcoin key management. In: Workshop on Usable Security (USEC). 2015.

[Ga05] Garfinkel, Simson L; Margrave, David; Schiller, Jeffrey I; Nordlander, Erik; Miller, Robert C: How to make secure email easier to use. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, S. 701–710, 2005.

[Ge] Gervais, Arthur; Ritzdorf, Hubert; Karame, Ghassan O; Capkun, Srdjan: Tampering with the delivery of blocks and transactions in bitcoin. Bericht, Cryptology ePrint Archive, Report 2015/578, 2015. http://eprint. iacr. org.

[GFFK06] Gaw, Shirley; Felten, Edward W; Fernandez-Kelly, Patricia: Secrecy, flagging, and paranoia: adoption criteria in encrypted email. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, S. 591–600, 2006.

[GKL15] Garay, Juan; Kiayias, Aggelos; Leonardos, Nikos: The bitcoin backbone protocol: Analysis and applications. In: Advances in Cryptology-EUROCRYPT 2015, S. 281–310. Springer, 2015.

[GM05] Garfinkel, Simson L; Miller, Robert C: Johnny 2: a user test of key continuity management with S/MIME and Outlook Express. In: Proceedings of the 2005 symposium on Usable privacy and security. ACM, S. 13–24, 2005.

[Go] Goldfeder, Steven; Gennaro, Rosario; Kalodner, Harry; Bonneau, Joseph; Kroll, Joshua; Felten, Edward W.; Narayanan, Arvind: , Securing Bitcoin wallets via a new DSA/ECDSA threshold signature scheme. Accessed 2015-06-09.

[He15] Heilman, Ethan; Kendler, Alison; Zohar, Aviv; Goldberg, Sharon: Eclipse Attacks on Bitcoin's Peer-to-Peer Network. In: 24th USENIX Security Symposium (USENIX Security 15). USENIX Association, Washington, D.C., S. 129–144, August 2015.

[Sh06] Sheng, Steve; Broderick, Levi; Koranda, Colleen Alison; Hyland, Jeremy J: Why johnny still can't encrypt: evaluating the usability of email encryption software. In: Symposium On Usable Privacy and Security. 2006.

[Wh94] Wharton, Cathleen; Rieman, John; Lewis, Clayton; Polson, Peter: The cognitive walkthrough method: A practitioner's guide. In: Usability inspection methods. John Wiley & Sons, Inc., S. 105–140, 1994.

[WT99] Whitten, Alma; Tygar, J Doug: Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0. In: Usenix Security. Jgg. 1999, 1999.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 149

Obtaining personal data and asking for erasure: Do app vendors and website owners honour your privacy rights?

Dominik Herrmann[1], Jens Lindemann[2]

Abstract: EU Directive 95/46/EC and the upcoming EU General Data Protection Regulation grant Europeans the right of access to data pertaining to them. Consumers can approach their service providers to obtain all personal data stored and processed there. Furthermore, they can demand erasure (or correction) of their data. We conducted an undercover field study to determine whether these rights can be exerted in practice. We assessed the behaviour of the vendors of 150 smartphone apps and 120 websites that are popular in Germany. Our deletion requests were fulfilled in 52 to 57 % of the cases and less than half of the data provision requests were answered satisfactorily. Further, we observed instances of carelessness: About 20 % of website owners would have disclosed our personal data to impostors. The results indicate that exerting privacy rights that have been introduced two decades ago is still a frustrating endeavour most of the time.

Keywords: Germany, Selbstauskunft, disclosure, deletion, empirical, websites, apps, GDPR

1 Introduction

Consumers are losing control of their personal data due to its pervasive collection in online services and apps. Answering the simple question "Who knows what about me?", which is at the core of informational self-determination, has become a complex and involving task. However, users *are* interested in answers. For instance, Lightbeam (previously called Collusion), which visualizes cookie-based tracking efforts, has been downloaded more than 600,000 times [Mo15]. In a recent survey, more than 90 % of respondents stated that it was important for them (a) to be in control about *who* gets information about them and (b) to control *what* information is collected [Pe15]. Researchers have proposed *privacy agents* for this purpose, client-side tools that keep track of the data entered into form fields [KNP10]. However, privacy agents have not seen widespread adoption so far.

EU Directive 95/46/EC [Eu95] and the upcoming General Data Protection Regulation (GDPR) grant citizens the right to obtain all data relating to them from data controllers. Citizens can also demand rectification, erasure or blocking of their data. In Germany, these provisions have been implemented in the Federal Data Protection Act in 2009 (cf. http://gesetze-im-internet.de/englisch_bdsg/). They are mostly exercised for scoring (e. g. the German "Schufa") and marketing services (e. g. Arvato) at the moment. In principle, consumers can also exercise them for all online service providers, but their effectiveness

[1] Universität Siegen, Institut für Wirtschaftsinformatik, Lehrstuhl für IT-Sicherheitsmanagement, Kohlbettstr. 15, 57072 Siegen, herrmann@wiwi.uni-siegen.de
[2] Universität Hamburg, Fachbereich Informatik, Arbeitsbereich Sicherheit in verteilten Systemen, Vogt-Kölln-Str. 30, 22527 Hamburg, jens.lindemann@informatik.uni-hamburg.de

is unknown so far. Will ordinary online shops bother to dig into their database when asked via e-mail by a customer?

Until now there was only circumstantial evidence suggesting that many data controllers are reluctant to answer data provision enquiries and that they fail to carry out deletion requests. Moreover, we wanted to evaluate whether data controllers handled personal data with due diligence. Therefore, we conducted two studies analysing the behaviour of vendors of services that are popular in Germany, comprising 150 smartphone apps and 120 websites. Both datasets have been compiled in the months of August and September of 2014.

This paper is structured as follows. First, we review related work in Sect. 2. In Sect. 3 we describe our methodology and present the results obtained for smartphone app vendors. Results of our experiments with website owners follow in Sect. 4. We discuss the results in Sect. 5, before we conclude the paper in Sect. 6.

2 Related work

We are not aware of previous research studying the behaviour of online service providers to determine the effectiveness of the right of access to personal data in practice. Balebako et al. [Ba14] analysed the privacy attitudes and the behaviour of app developers. However, in contrast to our undercover study, they conducted interviews and used an online survey. Most research about online privacy focuses on software, e. g. app permissions as well as what kind of data is being collected by apps and whom it is shared with. Another line studies legal and usability aspects, for instance by analysing privacy policies. In the following we will review recent work along these lines.

Zang et al. [Za15] monitored HTTP(S) transmissions of popular apps and found that many apps share information with third-party websites. TaintDroid is an effort to detect privacy violations by means of taint analysis [En10]. The second GPEN Privacy Sweep [Of14] conducted by privacy enforcement authorities found that the majority of apps does not provide sufficient information for the user to understand why it is necessary to grant the requested permissions, which were found to be excessive in relation to the functionality for 31 % of the apps. Other studies are concerned with the willingness of smartphone users to grant apps certain permissions and/or their willingness to share personal information with third parties [Li12, LLS14]. ICSI Haystack (https://www.haystack.mobi) alerts users about data leaks and collects data for research on privacy in mobile ecosystems.

Sunyaev et al. [Su15] studied the 600 most popular mobile health apps and found that only about 30 % did have a privacy policy, out of which about two thirds “did not specifically address the app itself”. They also deem these policies to be hard to read. Balebako et al. [Ba15] studied how well users will take notice of the privacy policy of an app depending on when it is being displayed. They found displaying the notice within the app to be more effective than displaying it in the app store. Schaub et al. [Sc15] analyse how privacy notices should be designed to be most effective, i. e. be noticed by users. Vila et al. [VGM04] state that users do not pay attention to privacy policies, as they are not a reliable indicator for whether the user’s privacy will actually be respected by a website. A similar observation

has been made by Chia et al. [CYA12] for mobile apps: they found no reliable indicator for whether an app would actually respect a user's privacy. Pollach [Po07] also deems privacy policies to be ineffective in addressing user's privacy concerns due to them mostly being designed "with the threat of privacy litigations in mind". Through a linguistic analysis, she found that questionable uses of data are often downplayed in policies. Despite users currently not paying attention to privacy policies, there is evidence that making the level of privacy easily visible to potential customers (e. g. by presenting it in a table) can influence their purchase decision, with them being willing to pay a premium for privacy [Ts11].

So far, there is only little work in practice. The browser add-on ToS;DR (https://tosdr.org) enables users to better understand privacy policies. For selected sites, it displays a summarised version of the terms of service at the click of a button and provides a colour-coded rating. The website *selbstauskunft.net* offers to send personal data enquiries to companies and authorities for German consumers, requiring virtually no effort. On *justdelete.me*, consumers can find information on how to delete accounts with many online services.

3 First study: popular smartphone apps

3.1 Data collection

For the selection of popular apps we relied on AppAnnie (http://www.appannie.com/), a market research company that monitors the downloads of apps from Apple's and Google's stores. We downloaded AppAnnie's list of the 500 most popular apps (in terms of downloads in Germany). From this list we took the 25 most popular free as well as the 25 most popular paid apps each for Android and iOS (100 apps in total). In addition, we randomly selected 50 free apps per platform from apps ranked between 50 and 500 in order to extend the scope of the survey. If a specific app was available for both platforms, precedence was given to the Android version. The selected apps are listed in Appendix 1.

In the following we characterize the dataset (see Fig. 5 in the appendix). Entertainment apps and games dominate the dataset with a share of 46 % of all apps, followed by apps that are used for work and information exchange (39 %). The smallest groups are apps used for communication and shopping (8 % and 7 %, respectively). Most app vendors are located in Germany (38 %). Vendors from other European countries are responsible for 21 % of the apps in the dataset, while 36 % are from other parts of the world (mainly the US). For 5 % of the apps, it was not possible to determine the vendor's residence. While it is beyond the scope of this paper to profoundly assess the legal situation, we note that Section 1 of the German Federal Data Protection Act states that it *is* applicable to vendors from non-EU countries under certain circumstances. While it may be difficult to exercise privacy rights against vendors in foreign countries, we still include those vendors in our study in order to compare their behaviour with that of domestic vendors.

We also tried to roughly categorize the vendors in terms of their size based on publicly available information, such as the number of employees. We found that 41 % of vendors are small, i. e. they consist of a single developer only, about a third are businesses with multiple developers, and 25 % are large enterprises.

3.2 Methodology for interaction with app vendors

The first objective of the study is to assess how app vendors react when users contact them to request (a) provision of their personal data and (b) deletion of their personal data. In order to obtain realistic results, we did not want to disclose to them that we were conducting a study about their behaviour. Instead, we approached the vendors like ordinary consumers. The second objective is to validate the claims received by the vendors.

We downloaded each app from the app store onto a smartphone and used it for a couple of minutes. When an app allowed to create an account or asked to provide personal data, we entered as many details as possible. We made the following arrangements to avoid arousing suspicion. For each app, we set up a fake profile having a randomly chosen German name and a random birth date. When asked to choose login credentials, we derived the username from the name and year of birth, while the password was chosen randomly. For the postal address, we used that of the Computer Science Department of University of Hamburg, which provides a satisfactory cover because it also hosts a number of other companies. We instructed the campus post office to accept mail for our fake identities, even though it was not addressed to the Computer Science Department. Moreover, we registered the domain barmail.de and used e-mail addresses of the form *firstname.lastname@barmail.de*.

We made a note of what data we entered in each app. Furthermore, we monitored the traffic of all apps to determine what pieces of data were disclosed to the vendors. For this purpose, we ensured that the apps were used while the smartphone was on a WiFi connection. We installed a custom CA root certificate on the phone to record all HTTP(S) traffic with BurpSuite (http://portswigger.net/burp). When we were analysing the responses of the vendors at a later stage, the recorded traffic allowed us to **validate the claims** of the vendors regarding what pieces of information they have or have not received about our account and whether or not their apps were transferring sensitive pieces of data (e. g. IMSI/IMEI) to other parties.

One week later, we approached each vendor via e-mail. This **first data provision enquiry** consisted of an informal e-mail in German or English (depending on the language of the app). The mail was sent from the address used during account setup. If no account was set up in the app, the mail was sent from the e-mail address used to log in to the Apple iTunes or Google Play store. We sent the e-mail to the address stated on the app page within the respective store. If no address was stated there, we looked up a contact address on the vendor's website or submitted our enquiry via a web form provided there. In the enquiry we stated the respective e-mail address and username and asked the vendor to provide all personal data that was stored on its servers and the purpose of their collection.

If there was no response within one week or if it was not satisfactory (cf. Sect. 3.3), we sent a **second data provision enquiry**. The second enquiry was written more formally than the first one, because we hypothesized that a more formal and threatening tone would illicit more responses. Therefore, the enquiry contained an explicit reference to Section 34 of the German Federal Data Protection Act and in closing we announced that we would notify the supervisory authority according to Section 38 of the aforementioned act (which we did not in fact do) if we did not receive a response within seven days.

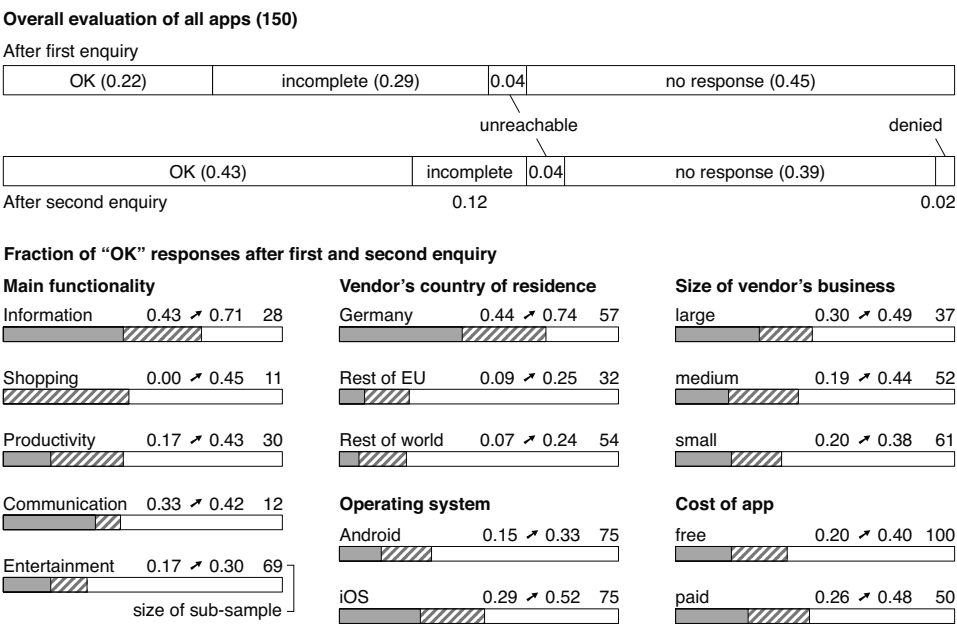


Fig. 1: Results for data provision enquiry for popular apps

Four weeks after the last contact, we continued with the **analysis of account deletion practices**. This analysis was limited to those 56 apps allowing to set up an account. First, we tried to delete the account from within each app and on the vendor's website on our own. Otherwise we sent a **first deletion request** to the vendor from the e-mail address used during registration, in which we informally asked to have our account deleted. If we did not receive a response within one week or if we determined that the account was not deleted completely, we followed up with a formal **second deletion request**, referring to Section 35 of the Data Protection Act. Again, we announced that we would notify the supervisory authority unless we received a response within seven days. Obviously, it is impossible for consumers to verify that all of their personal data has been deleted completely by a vendor. Therefore, we can only look for signs that indicate an *incomplete* deletion. We **verified claims about the purported deletion** of an account as follows: First, we tried to log in with the previously used credentials, which should not be possible any more. Second, we tried to reset the password, which, if successful, indicates that the account had been disabled rather than deleted. Third, we tried to create a new account with the same credentials. This should succeed. Otherwise, the vendor keeps some records about "deleted" accounts (which may in fact be compliant with the data protection act).

3.3 Analysis and results for apps

The results for the data provision enquiry are summarised in Fig. 1. The upper part shows the overall distribution of responses. A satisfactory **(OK)** response was scored if the response contained the actual data values that we had observed being transferred from the

app to the vendor, or, in case we did not observe the app transmitting anything to the vendor at all, if the response stated this fact. The first enquiry was answered satisfactorily by only 22 % of vendors, increasing to 43 % after the more formal second enquiry. Two of the vendors responded by postal mail, all others via e-mail. Some vendors only stated what kinds of information they typically store in an abstract fashion, i. e. they had not looked up our entry in their records at all. Others referred us to their privacy policy. Those cases were recorded as **incomplete** responses, because they did not provide our personal data as requested. The fraction of incomplete responses decreased from 29 % (first enquiry) to 12 % (second enquiry). This indicates that customers have to be insisting to obtain the desired information. For 4 % of the apps, we found no way to contact the vendor or our e-mail bounced **(unreachable)**. Many vendors did not reply to our mails (45 and 39 %, respectively). In response to the second enquiry, 2 % of vendors explicitly **denied** to answer our enquiry by e-mail, instructing us to call them or to send our enquiry by postal mail.

The bottom part of Fig. 1 focuses on the "OK" cases. The highest fraction of satisfactory responses is achieved by German vendors (44 % and 74 %, respectively). One reason for this discrepancy may be that vendors from Germany are more sensitive to privacy than vendors from other countries. A reason for their poor performance after the second enquiry may be that *foreign* vendors do not fear the consequences of customers notifying the German supervisory authorities, which cannot exert power in foreign jurisdictions. Vendors of apps that are mainly used for providing or exchanging information scored better (71 %) than those of other apps (e. g. games and entertainment apps: 30 %). Further, vendors of iOS apps provided more satisfactory responses than those of Android apps (52 vs. 31 %, respectively). Vendors of paid apps scored only marginally better than those of free apps (48 vs. 40 %). Finally, large businesses scored best (49 %).

Figure 2 presents the results obtained for our deletion requests sent to the vendors of the subset of 56 apps in which we had created a user profile that was stored on the vendor's servers. We could delete 13 accounts on our own, either from within the app or on the vendor's website. Overall, deletion succeeded considerably more often than obtaining personal data. After the first informal request, 20 accounts were **completely deleted**. Out of the 13 accounts mentioned above, 10 were completely deleted, resulting in a total of 54 % of accounts (compared to 22 % satisfactory responses after the first data provision enquiry). The second request increased this fraction only slightly to 57 %. All of our e-mails were delivered correctly this time, but in the end there was still **no reaction** in 23 % of all cases. In 5 cases (9 %; including 3 of the 13 accounts we were able to delete on our own), we were able to revive a "deleted" account **(deletion failed)**. The affected vendors declined to completely delete the accounts even after we confronted them with our finding (still using the fake profile as a cover). In 11 % of cases deletion was **impossible**: in half of these cases we couldn't find a way to contact the vendor, in the other cases support staff claimed that it was impossible to delete the account.

It may be perceived positively that all it takes for consumers to delete their account is writing an e-mail. Disturbingly, some vendors carried out deletion instantaneously, failing to check the identity of the sender. This lapse allows malicious users to delete the accounts of others by sending faked deletion requests with a forged sender address.

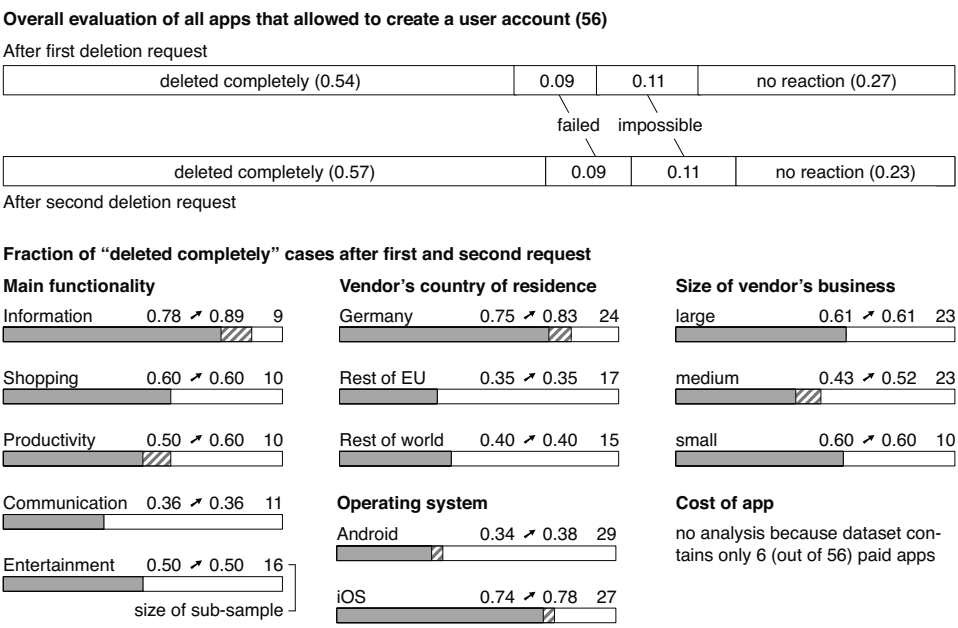


Fig. 2: Results for deletion request for popular apps

4 Second study: popular websites

4.1 Data collection

The dataset used in the second study contains popular websites that allow their users to create a user account to store personal data. The dataset was compiled based on the Alexa Top List for Germany (http://www.alexa.com/). First, we visited the 100 most popular websites. 57 of those 100 sites allowed us to set up an account and were therefore added to the dataset. Second, we randomly sampled 100 sites from the websites with ranks between 100 and 500. We added 63 of those sites to the dataset. Thus, the dataset contains 120 websites that are popular in Germany and allowed us to set up a user account (cf. Appendix 1). Some of the selected sites belong to vendors that have already been studied in the mobile app dataset. For these sites we duplicated the result obtained in the first study in order to avoid bias due to sending multiple queries to a vendor within a short period of time. In contrast to the app dataset, shopping sites make up the largest part (40 %, compared to 7 % of apps), while the fraction of entertainment sites is quite small (12 %, compared to 46 % of apps). Further, there are much more large businesses in the website dataset (46 %, compared to 25 % of apps). Figure 6 in the appendix shows the distributions in detail.

4.2 Methodology for interaction with website owners

The methodology used to study the behaviour of website owners resembles the procedure used for apps (cf. Sect. 3.2), i. e. we created a plausible fake user profile on each website,

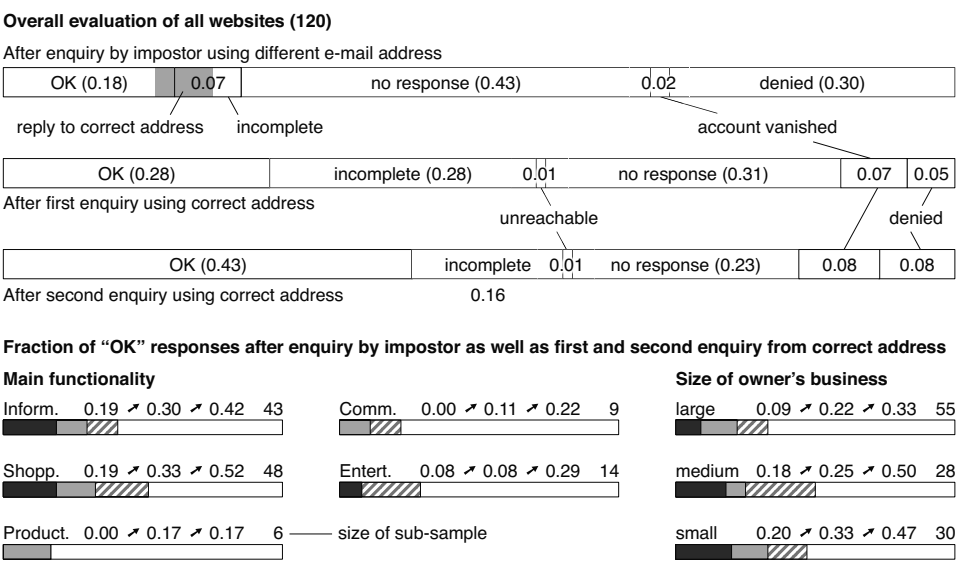


Fig. 3: Results for data provision enquiry for popular websites

using e-mail addresses of the form *firstname.lastname@barmail.de*. However, there is one important deviation: This time we also wanted to study how diligently site owners handled data provision enquiries. After all, the responses contain sensitive personal data. An impersonator should not be able to obtain personal data of a different person by fooling a website owner with a carefully crafted data provision enquiry. Spurious requests should be ignored and replies should only be sent to the e-mail or postal address stored in the user account. For this analysis we registered the domain fair-konsult.de, which was to serve as a cover for a purported impostor. The impostor would send a data provision enquiry from the "office address" *firstname.lastname@fair-konsult.de*. The text of the enquiry did not mention this discrepancy. Instead, it was an ordinary informal enquiry, i. e. it mentioned the *firstname.lastname@barmail.de* address (and username, where appropriate) and stated the desire to obtain the personal data stored under that account.

If we received no satisfactory response from a site owner, we repeated the enquiry from the correct address, in analogy to the procedure used for apps. This time we created a second fake account on each site that was used for the deletion request. Having seen the results of the first study, we were concerned that our earlier provision enquiry might be stored in the customer support software of the site owner and bias the outcome of our deletion request.

4.3 Results

The results for the data provision enquiry are summarised in Fig. 3. 30 % of site owners noticed the incorrect e-mail address and responded with an explicit denial, while most (43 %) did not bother replying to our bogus enquiry at all. However, most interestingly, 25 % of website owners were indeed fooled into responding to our impostor with a satisfactory

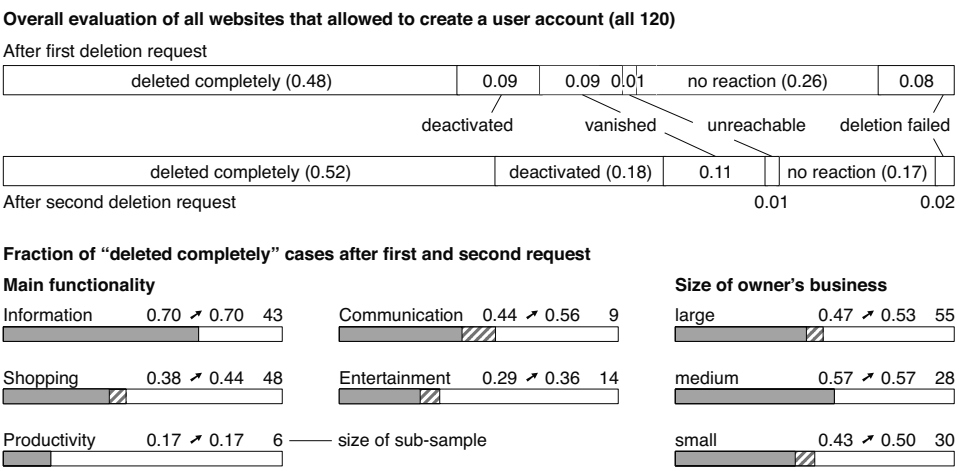


Fig. 4: Results for deletion request for popular websites

or incomplete response. One fifth of those replied to the *barmail.de* address; the majority *would* have disclosed sensitive personal data to an impostor, though. The breakdown of the results shows that this kind of fault happened at businesses of all sizes. Including the "OK" cases obtained with the impostor's enquiry, 28 % of site owners have provided a complete and correct response to the first enquiry. Eventually, 43 % of responses were satisfactory. These and most of the remaining figures are in line with the results obtained for apps (cf. Sect. 3.3). However, large businesses do not perform better this time. Moreover, some owners said they could not provide any data, because the account did not exist any more (8 % of accounts had **vanished** by the time of the second enquiry). In less than a handful of cases, site owners required the customer to call them to process the enquiry. As in the app study, only two site owners responded by postal mail.

Figure 4 presents the results regarding the deletion requests. After the second request, 52 % of site owners had completely deleted our account. In contrast to the app study, some vendors (18 %) claimed that they could not delete the account for regulatory reasons (e. g. taxation). They promised to block the personal data. Again, a small fraction of site owners (11 %) claimed that there were no traces of the account in question (about eight weeks after it had been created). Apparently, some site owners purge inactive accounts after a while. This is in fact beneficial for privacy as it reduces the risk of data leaks for users.

5 Discussion

A typical ethical issue in undercover field research is that it consumes resources of the studied subjects without their consent, i. e. in our case the time of the employees of the surveyed service providers. However, we believe that this approach is the only way to investigate the problems related to data provision enquiries and deletion requests in practice. The insights gained may create awareness and sketch a path of action for regulatory authorities. In the design of our studies, we tried to minimize the burden on vendors.

While our objective was to survey popular apps and websites, we stress that the two samples were not meant to be representative. Therefore, the results cannot be generalized to the whole population of apps and websites. Further, the results of the first study may be biased due to giving preference to the Android version of an app in case it happened to be selected for both platforms. This means that apps that were very popular on both platforms have only been surveyed on Android, leading to the surveyed iOS apps being slightly less popular on average. Another potential problem is that it was classified as a failure to respond to our first enquiry if the response did not come within a week. As some responses arrived after this time frame had passed, it might be more appropriate to wait for a longer time in future studies. We are currently in the process of conducting a more extensive study, based on a more controlled and larger sample of smartphone apps.

Another potential source of bias stems from the design of the second study (websites), which includes an additional step to assess the viability of social engineering attacks. This additional query may have influenced the response of vendors. First, vendors that need considerably more time than we expected may have received our authentic mail (sent in the second step) *before* they were about to reply to the faked mail. Second, vendors that have refrained from replying to our faked mail may actually have an increased inclination to respond once they receive our second mail (because it is the second contact for the same user account). Thus, these results are biased to some degree. In our follow-up study, we will therefore leave out the steps to analyse the viability of social engineering.

As noted in Sect. 3.2, it is impossible for us to verify whether a vendor has completely deleted personal information. This is why we have used techniques to infer whether any information is left behind. Another approach would have been to send another data provision enquiry after the deletion request. It would be interesting to see whether vendors would really admit that they have not complied with a deletion request. We leave this question for future work.

6 Conclusion

In this paper we quantified the effectiveness of the right of access to personal data with two studies. We found that for both apps and websites, only 43 % of our data provision requests were answered satisfactorily by vendors. Many of those who provided satisfactory information failed to do so after the first informal enquiry. Accordingly, obtaining one's personal data often requires perseverance and precise knowledge of the legal code. This may discourage many citizens from exercising this right. For deletion requests, the situation is somewhat better: Between 52 % to 57 % of accounts were deleted completely. This time, a large majority of vendors acted upon the first request, which may be due to the little effort involved. Worryingly, about 20 % of website owners were found to be susceptible to social engineering and provided the requested data to a different e-mail address than the one used to register the account. Although the right of access has been introduced about two decades ago, exercising it is both ineffective and inefficient. Instead of error-prone manual processing of such requests, service providers should (be legally bound to) offer consumers a standardised interface to access, rectify and erase/block their personal data.

References

[Ba14] Balebako, Rebecca; Marsh, Abigail; Lin, Jialu; Hong, Jason; Cranor, Lorrie F.: The Privacy and Security Behaviors of Smartphone App Developers. In: USEC. 2014.

[Ba15] Balebako, Rebecca; Schaub, Florian; Adjerid, Idris; Acquisti, Alessandro; Cranor, Lorrie: The Impact of Timing on the Salience of Smartphone App Privacy Notices. In: SPSM '15. ACM, pp. 63–74, 2015.

[CYA12] Chia, Pern Hui; Yamamoto, Yusuke; Asokan, N.: Is this App Safe? A Large Scale Study on Application Permissions and Risk Signals. In: WWW 2012. pp. 311–320, 2012.

[En10] Enck, William; Gilbert, Peter; Chun, Byung-Gon; Cox, Landon P.; Jung, Jaeyeon; McDaniel, Patrick; Sheth, Anmol: TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In: OSDI 2010. pp. 393–407, 2010.

[Eu95] European Parliament: Directive 95/46/EC. Official Journal of the European Communities No L 281 (38), 31–50, 1995.

[KNP10] Kolter, Jan; Netter, Michael; Pernul, Günther: Visualizing Past Personal Data Disclosures. In: ARES 2010. pp. 131–139, 2010.

[Li12] Lin, Jialiu; Sadeh, Norman M.; Amini, Shahriyar; Lindqvist, Janne; Hong, Jason I.; Zhang, Joy: Expectation and purpose: understanding users' mental models of mobile app privacy through crowdsourcing. In: Ubicomp 2012. pp. 501–510, 2012.

[LLS14] Liu, Bin; Lin, Jialiu; Sadeh, Norman M.: Reconciling mobile app privacy and usability on smartphones: could user privacy profiles help? In: WWW 2014. pp. 201–212, 2014.

[Mo15] Mozilla: Lightbeam for Firefox. https://addons.mozilla.org/en-US/firefox/addon/lightbeam/, Accessed Dec 9, 2015.

[Of14] Office of the Privacy Commisioner of Canada: Results of the 2014 Global Privacy Enforcement Network Sweep. https://www.priv.gc.ca/media/nr-c/2014/bg_140910_e.asp, September 2014.

[Pe15] Pew Research Center: Americans' Views About Data Collection and Security. http://www.pewinternet.org/2015/05/20/americans-views-about-data-collection-and-security/, May 2015.

[Po07] Pollach, Irene: What's wrong with online privacy policies? Comm. ACM, 50(9):103–108, 2007.

[Sc15] Schaub, Florian; Balebako, Rebecca; Durity, Adam L.; Cranor, Lorrie Faith: A Design Space for Effective Privacy Notices. In: SOUPS 2015. USENIX, pp. 1–17, July 2015.

[Su15] Sunyaev, Ali; Dehling, Tobias; Taylor, Patrick L; Mandl, Kenneth D: Availability and quality of mobile health app privacy policies. Journal of the American Medical Informatics Association, 22(e1):e28–e33, 2015.

[Ts11] Tsai, Janice Y.; Egelman, Serge; Cranor, Lorrie Faith; Acquisti, Alessandro: The Effect of Online Privacy Information on Purchasing Behavior: An Experimental Study. Information Systems Research, 22(2):254–268, 2011.

[VGM04] Vila, Tony; Greenstadt, Rachel; Molnar, David: Why We Can't Be Bothered to Read Privacy Policies - Models of Privacy Economics as a Lemons Market. In: Economics of Information Security, volume 12 of AICT, pp. 143–153. Springer, 2004.

[Za15] Zang, Jinyan; Dummit, Krysta; Graves, James; Lisker, Paul; Sweeney, Latanya: Who Knows What About Me? A Survey of Behind the Scenes Personal Data Sharing to Third Parties by Mobile Apps. http://techscience.org/a/2015103001, Oct. 30, 2015.

Appendix 1: Description of datasets

Due to space constraints the selected apps and websites are only available in the preprint version of this paper. It can be downloaded from https://svs.informatik.uni-hamburg.de/publications/2016/2016-02-04-PrivacyEnquiryPreprint.pdf or from arXiv.

Category in app store

M	C	S	N	W	M	Traffic	Travel	Health	C & S	Tools	Entertainment & Games
4	4	4	4	5	6	9	9	9	13	22	61

Main functionality

Shops	Comm.	Information	Productivity	Entertainment
11	12	28	30	69

Vendor's country of residence

?	other EU Countries	Other	Germany
7	32	54	57

Size of vendor's business

Large	Medium	Small
37	52	61

Category abbreviations: Marketplace, Community, Shopping, News, Weather, Music, Communications & Social

Fig. 5: Popular apps dataset overview; figures refer to number of apps out of 150 apps in total

Category

C	D	SN	Porn	Travel	Ent.	Misc.	Inform.	News	Community	Marketplace	Online Shop
5	5	5	7	7	7	8	10	14	15	15	22

Main functionality

Prod.	Com.	Entertainm.	Information	Shopping
6	9	14	43	48

Size of owner's business

Large	Medium	Small	?
55	28	30	7

Category abbreviations: Cloud Service, Dating, Social Networks, Entertainment, Miscellaneous, Information Service

Fig. 6: Popular websites dataset overview; figures refer to number of sites out of 120 sites in total

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 161

IT-Grundschutz in großen Institutionen – Herausforderungen und Lösungsstrategien

Timo Brecher[1], Sabine Kammerhofer[2], Severin Rast[3]

Abstract: Große Institutionen sind einerseits auf eine standardisierte Vorgehensweise für das Informationssicherheitsmanagement angewiesen, haben andererseits aber stets besondere Anforderungen, die vom Standard abweichen. Dies führt regelmäßig zu Herausforderungen bei der Anwendung des IT-Grundschutzes: Es gibt einen vielfältigen methodischen Kontext; der Informationsverbund ist sehr komplex; Projekte haben sehr lange Laufzeiten; es bestehen besondere betriebliche Anforderungen und oft die Notwendigkeit eigene Bausteine zu definieren. Dieser Artikel betrachtet diese Herausforderungen und stellt geeignete Lösungsansätze vor.

Keywords: Modellierung und Verifikation von Sicherheit, Sicherheitsmanagement, IT-Grundschutz, IT-Sicherheitskonzeption

1 Einleitung

Große Behörden und Unternehmen haben besondere Anforderungen an Planung, Realisierung und Betrieb zeitgemäßer und wirtschaftlicher IT-Systeme. Sicherheit ist hierbei ein wesentlicher Aspekt, der jedoch nie allein, sondern stets eingebettet in die Gesamtheit der Prozesse und Aktivitäten der Institution zu betrachten ist. Der IT-Grundschutz bietet hierzu eine bewährte Methodik und solide Maßnahmenkataloge, bringt jedoch in Anwendungsfällen abseits des idealen Informationsverbundes seine eigenen Herausforderungen mit.

2 Herausforderungen bei der Anwendung des IT-Grundschutzes in großen Institutionen

Im Rahmen der Anwendung der IT-Grundschutz-Methodik in verschiedenen großen Institutionen ließ sich eine Reihe von Problemstellungen identifizieren, die regelmäßig zu bewältigen sind. In den nachfolgenden Abschnitten werden fünf wesentliche Herausforderungen näher betrachtet.

[1] Infodas GmbH, IT Security, Rhonestr. 2, 50765 Köln, t.brecher@infodas.de
[2] Infodas GmbH, IT Security, Rhonestr. 2, 50765 Köln, s.kammerhofer@infodas.de
[3] Infodas GmbH, IT Security, Rhonestr. 2, 50765 Köln, s.rast@infodas.de

2.1 IT-Grundschutz im Kontext der IT-Governance

Der mit der IT-Grundschutz-Vorgehensweise beschriebene Anwendungsansatz für die Etablierung und Aufrechterhaltung eines Managementsystems für Informationssicherheit und Erstellung von IT-Sicherheitskonzepten ist vollständig kompatibel zum internationalen Standard ISO/IEC 27001 und ist gut auf den Einsatz in Behörden und Unternehmen ausgerichtet. Die IT-Grundschutz-Kataloge sind seit Jahren praxiserprobt und ermöglichen eine effiziente Auswahl der relevanten IT-Sicherheitsmaßnahmen. In großen Institutionen entstehen bei der Einführung des IT-Grundschutzes allerdings stets Wechselwirkungen mit anderen Standards und Vorschriften. Neben internen Vorschriften zur IT-Sicherheit[4], können dies zum einen interne und externe Anforderungen und Standards aus den Bereichen Datenschutz[5], Geheimschutz[6], Risikomanagement[7] sein; zum anderen spielen Architekturmodelle[8] und Best Practise Frameworks[9] in der modernen IT-Governance eine immer größere Rolle. Dies führt zu einer Vielfalt an unterschiedlichen Betrachtungsweisen für die in einer Institution eingesetzte IT, die jeweils mit eigenen Begriffsdefinitionen und analytischen Strukturen einhergehen. Für ein funktionierendes ISMS und ein wirksames IT-Sicherheitskonzept ist ein einheitliches institutionsübergreifendes Verständnis des Informationsverbundes und der zu seiner Realisierung genutzten IT jedoch eine wesentliche Voraussetzung. Im Rahmen einer transparenten IT-Sicherheitskonzeption müssen daher die für eine Institution relevanten Betrachtungsweisen für alle Zielgruppen nachvollziehbar miteinander in Beziehung gesetzt werden.

2.2 IT-Sicherheitskonzeption für komplexe Informationsverbünde

Die in der Keynote zum BSI-Kongress präsentierte Bestandsaufnahme zeigt, dass die IT der unmittelbaren Bundesverwaltung Ende 2013 mit ihren knapp 200 Behörden auf über 1.300 Rechenzentren und Serverräume an unzähligen Standorten verteilt ist [MAIZ]. Bei großen Bundesbehörden besteht somit häufig das Problem, dass der zu betrachtende Anwendungsbereich bzw. Informationsverbund zu umfangreich und zu komplex ist, um diesen über ein einzelnes IT-Sicherheitskonzept adäquat abdecken zu können. Dies führt zu einer Unterteilung des Informationsverbundes in einzelne Teilbereiche. Die wesentlichen Herausforderungen stellen hier die unterschiedlichen Modellierungsmöglichkeiten sowie der Umgang mit Redundanzen (z.B. gleiche Maßnahmen in gleichen Bausteinen unterschiedlicher Teilbereiche), die in weiterer Folge zu Konsistenzproblemen führen können.

Der Maßnahmenkatalog des IT-Grundschutzes wurde mit der Zielsetzung erstellt,

[4] Z.B. [ZDv 54/100]
[5] Z.B. [BDSG] oder [EU Rl]
[6] Z.B. [VSA] oder [GSH]
[7] Z.B. [MaRisk] oder [ISO31000]
[8] Z.B. [IETF] oder [SOA]
[9] Z.B. [ITIL] oder [COBIT]

Informationsverbünde von Unternehmen und Behörden als Ganzes zuverlässig zu schützen. Dabei werden die organisatorischen, technischen, personellen und infrastrukturellen Aspekte untersucht.

Beispiel: Mit Einführung des IT-Grundschutzes als Methodik für die Erstellung von IT-Sicherheitskonzepten in der Bundeswehr muss die Verzahnung von projektbezogenen IT-Sicherheitskonzepten und dienststellenbezogenen IT-Sicherheitskonzepten[10] grundlegend neu überdacht werden. Eine Unterscheidung zwischen Maßnahmen, die in der Verantwortung einer Dienststelle und Maßnahmen, die in Verantwortung des Projekts liegen, ist nicht vorgesehen. Diese Unterscheidung muss der Ersteller eines bundeswehrspezifischen IT-Sicherheitskonzepts vornehmen.

2.3 Besonderheiten bei langen Projektlaufzeiten

Ein weiteres Problem bei der Anwendung des IT-Grundschutzes auf Projekte in großen Institutionen stellt die oftmals lange Projektlaufzeit dar. Bei einer Projektlaufzeit von mehreren Jahren sind Dienstpostenwechsel, z.B. beim IT-Sicherheitsbeauftragten, keine Seltenheit, so dass im Anschluss eine Einarbeitungs- und Übergabezeit stattfinden muss, um einen gänzlichen Knowhow-Verlust zu vermeiden.

Lange Projektzeiten sind auch ein Problem, wenn sich in diesem Zeitraum Vorschriften- und Weisungslagen ändern. Wenn die Vertragsgestaltung diesen Aspekt nicht berücksichtigt, ergeben sich erhebliche Probleme bei der Zusammenarbeit mit den Herstellern/Dienstleistern. Falls durch die Änderung der Vorschriften- und Weisungslagen Mehraufwände entstehen, sind die Hersteller/Dienstleister in der Regel nicht in der Lage die IT-Sicherheitsmaßnahmen gemäß IT-Sicherheitskonzept kostenneutral umzusetzen, so dass vertragliche Änderungen notwendig werden, die zeitaufwendiges Änderungsmanagement und zusätzliche Nachweisführung erfordern.

Beispiel: Durch eine behördenspezifische Vorschriftenänderung kann ein System nur mit einem vorschriftenkonformen Patchmanagement abgenommen werden. Diese Anforderung wurde zwar von der Behörde im Lastenheft dokumentiert, ohne jedoch die zum Zeitpunkt des Vertragsschlusses noch nicht vorliegenden behördenspezifischen Anforderungen zu berücksichtigen. Eine Abnahme kann in diesem Fall nur erfolgen, wenn der Vertrag entsprechend angepasst wird, was oft zu erheblichen Mehrkosten führt.

2.4 Besondere betriebliche Anforderungen

Der IT-Grundschutz ist für die Anwendbarkeit sowohl auf Büro- als auch auf Rechenzentrumsumgebungen ausgelegt. Falls von der Institution an ein IT-System

[10] Im dienstellenbezogenen IT-Sicherheitskonzept wird der gesamte Informationsverbund der Dienststelle mit den darin genutzten bzw. betriebenen Projekten betrachtet.

besondere betriebliche Anforderungen gestellt werden, führt diese Pauschalisierung des IT-Grundschutzes zu Problemen, wenn die Umsetzung von IT-Sicherheitsmaßnahmen den besonderen betrieblichen Anforderungen widerspricht. Eine Entbehrlichkeit der IT-Sicherheitsmaßnahme kann nicht begründet werden. Damit ist eine Risikobewertung erforderlich.

Beispiel: Ein für den operativen Betrieb wichtiges Computersystem in der Brückenkonsole an Bord eines Schiffes soll die IT-Grundschutzmaßnahme M 4.2 – Bildschirmsperre [GSK] umsetzen. In dieser Maßnahme wird gefordert, dass der Zeitraum, nach dem sich eine Bildschirmsperre wegen fehlender Benutzereingaben aktiviert, gewisse Grenzen weder unter- noch überschreiten sollte. Die Umsetzung dieser Maßnahme ist jedoch in dieser spezifischen Einsatzumgebung nicht möglich, da der Blick auf die Konsolenanzeige niemals versperrt sein darf, um kurzfristig auf navigatorische Anweisungen reagieren zu können.

2.5 Erweiterung des IT-Grundschutzes durch zusätzliche Bausteine

„Die IT-Grundschutz-Kataloge enthalten die Gefährdungslage und die Maßnahmenempfehlungen für verschiedene Komponenten, Vorgehensweisen und IT-Systeme, die jeweils in einem Baustein zusammengefasst werden“ [100-2, S. 60]. Der Fortschritt der technischen Entwicklung einerseits und die besonderen Anforderungen großer Behörden und Unternehmen an ihre IT andererseits führen dazu, dass nicht für alle eingesetzten Hard- und Softwarekomponenten oder übergreifende Aspekte passende Bausteine in den IT-Grundschutz-Katalogen vorliegen. Insbesondere in den Bereichen von Appliances und Embedded Systems, neuen Softwareversionen mit deutlich erweitertem Funktionsumfang[11] oder auch besonderem Schutzbedarf ist die Erstellung neuer Bausteine notwendig[12]. Dies führt zu einer Erhöhung des Aufwands für die Erstellung des IT-Sicherheitskonzepts und zu Unsicherheiten in der Planung des IT-Sicherheitsmanagements, etwa bei Ergänzungslieferungen des IT-Grundschutzes, in denen offizielle Bausteine eingeführt werden, die selbst erstellte ersetzen. Ein strategisch und langfristig ausgerichtetes ISMS braucht daher Prozesse zur effizienten Erstellung und zum Management selbsterstellter Bausteine.

3 Lösungsstrategien

Zu den im vorherigen Abschnitt beschriebenen Herausforderungen werden nachfolgend Lösungsstrategien vorgestellt, die basierend auf praktischen Erfahrungen eine universelle Anwendung finden können.

[11] Z.B. beim Umstieg von Windows 7 auf Windows 8/8.1

[12] Eine Vorgehensweise hierzu ist in der unveröffentlichten BSI-Richtlinie "Erstellung eines Bausteins für die IT Grundschutz Kataloge" beschrieben.

3.1 Abbildung des IT-Grundschutzes auf weitere Frameworks und Modelle

Gültige Vorschriftenlage, Governance-Frameworks und der IT-Grundschutz betrachten den Informationsverbund aus sehr unterschiedlichen Perspektiven. Diese transparent aufeinander abzubilden bzw. miteinander in Beziehung zu setzen und ein einheitliches Verständnis dieser Beziehungen über den gesamten ISMS-Prozess zu gewährleisten, ist ein Schlüssel für die institutionsübergreifende Wirksamkeit eines ISMS. Hierzu hat es sich bewährt, frühzeitig ein institutionsspezifisches Modell zu entwickeln und dieses in der gesamten IT-Sicherheitsdokumentation konsistent anzuwenden. Die folgende Abbildung stellt ein vereinfachtes Beispiel für ein solches Modell dar.

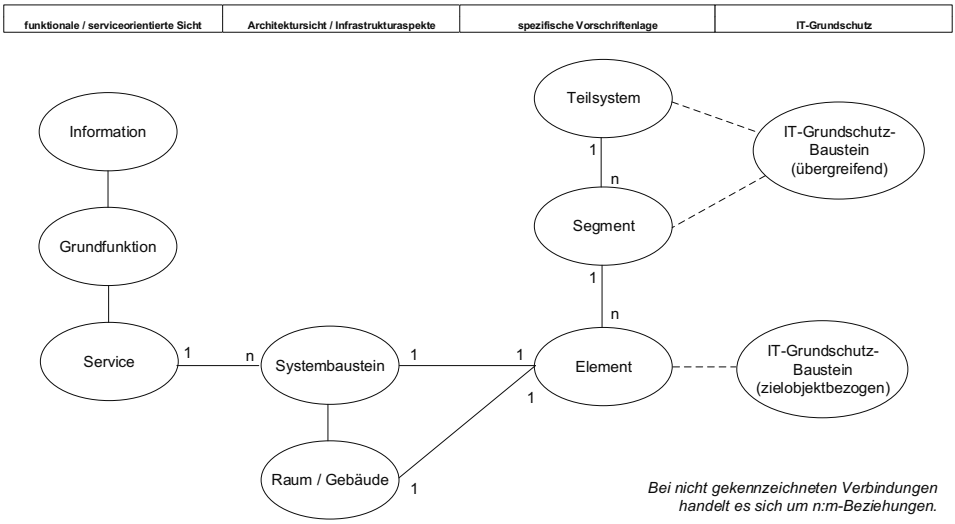


Abbildung 1: Abbildung unterschiedlicher Sichten auf den IT-Grundschutz

In der funktionalen Sicht eines serviceorientierten Frameworks werden die Services als grundlegende Elemente der Betrachtung identifiziert. Diese lassen sich Systembausteinen einer Architektursicht zuordnen, die wiederum als Elemente gemäß der spezifischen Vorschriftenlage der Institution[13] betrachtet werden. Diese Elemente werden als Zielobjekte mit IT-Grundschutz-Bausteinen der Schichten zwei bis fünf im IT-Sicherheitskonzept modelliert, während übergeordneten Struktureinheiten (Teilsysteme und Segmente) die IT-Grundschutz-Bausteine der Schicht eins zugeordnet werden.

[13] In diesem Beispiel [ZDv]

3.2 Modularisierung einer IT-Sicherheitskonzeption

Bei sehr umfangreichen und komplexen Institutionen hat sich im Rahmen der IT-Sicherheitskonzepterstellung ein modularer und schichtenorientierter Aufbau bewährt. Ziel der Modularisierung ist die redundanzfreie Zusammenfassung typischer Zielobjekte und ihrer korrespondierenden IT-Grundschutz-Bausteine bzw. sonstiger institutionsspezifischer Bausteine zu wiederverwendbaren, standardisierten Modulen des übergreifenden IT-Sicherheitskonzepts. Diese können bei der Erstellung von IT-Sicherheitskonzepten für konkrete Teilbereiche nach Bedarf genutzt werden.

Im Rahmen der Erstellung der übergeordneten IT-Sicherheitskonzeption werden neben den Modulen auch die organisatorischen Aspekte (insbesondere Schicht 1 und 2 der IT-Grundschutz-Kataloge des BSI) beschrieben. Untergeordnet werden verfahrens- und systembezogene IT-Sicherheitskonzepte erstellt, welche sich im Wesentlichen auf die Bausteine der Schichten 3 bis 5 beziehen und die bereits im übergreifenden IT-Sicherheitskonzept behandelten Themen größtenteils durch Verweise abdecken können.

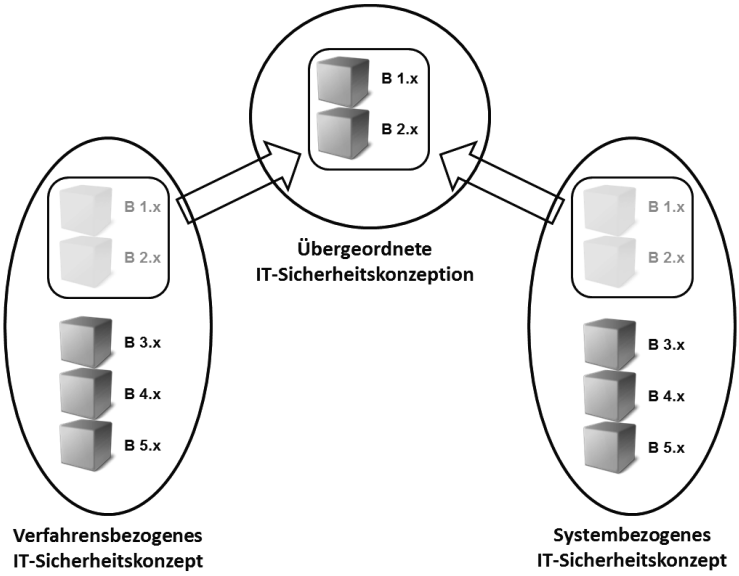


Abbildung 2: Referenzen von Modulen auf eine übergeordnete IT-Sicherheitskonzeption

Der modulare Aufbau der IT-Sicherheitskonzeption bietet die folgenden Vorteile:

- Wenn sowohl die Systemarchitektur als auch die IT-Sicherheitskonzepte modular strukturiert sind, können die System-Komponenten und -Bausteine auf einfache Weise den Modulen der IT-Sicherheitskonzepte zugeordnet werden (siehe auch Abschnitt 3.1).

- Bei neuen IT-Verfahren ist eine erneute Sicherheitsanalyse der schon bekannten und untersuchten Systemanteile in der Regel entbehrlich. Damit ist es ausreichend, nur noch die neu hinzukommenden Zielobjekte sicherheitstechnisch zu betrachten.

- Aufgrund des modularen Aufbaus der IT-Sicherheitskonzeption können Redundanzen in davon abgeleiteten Richtlinien und Anweisungen vermieden werden. So wird nicht nur die Erstellung der IT-Sicherheitskonzepte erleichtert, sondern insbesondere auch deren Fortschreibung vereinfacht, da Änderungen nur zentral notwendig sind und folglich auch Inkonsistenzen vorgebeugt wird.

- Durch den modularen Aufbau wird außerdem die Handhabbarkeit der IT-Sicherheitskonzepte wesentlich verbessert, da immer nur die jeweils relevanten Module der IT-Sicherheitskonzepte betrachtet werden müssen. Die hiermit verbundene klare Abgrenzung der Einzelaspekte erleichtert nicht nur die Abgrenzung von Zuständigkeiten bei der Bearbeitung, sondern auch die Durchführung von IT-Sicherheitsaudits.

3.3 Berücksichtigung von personellen und vertraglichen Änderungen

Bei langen Projektlaufzeiten ist ein funktionierendes Wissensmanagement bei Dienstpostenwechseln vorzusehen. Hierzu sind, neben dem Führen einer vollständigen Dokumentation, Stellvertreter von Schlüsselpersonal der IT-Sicherheit zu ernennen und von Beginn an in die Projektarbeit einzubinden, so dass bei Ausfall oder Wechsel von Personal die Aufgaben ohne eine lange Einarbeitungszeit übernommen werden können.

Bei der Planung von komplexen Realisierungsprojekten, bei denen Umsetzung von IT-Sicherheitsmaßnahmen gefordert ist, ist darauf zu achten, bereits bei Kalkulation und Vertragsgestaltung zukünftige Änderungen der Vorschriften- und Weisungslage zu berücksichtigen. Dafür hat sich eine den Baufortschritt begleitende und idealerweise vom realisierenden Auftragnehmer bzw. der realisierenden Stelle unabhängige Beratung durch Fachpersonal bewährt, die bereits in der Planungsphase beginnt. So können zukünftige Änderungen der Vorschriften- und Weisungslage, die Auswirkungen auf Anforderungen der IT-Sicherheit haben, kurzfristig berücksichtigt werden, um einen zeitgerechten Abschluss des Projektes nicht zu gefährden.

3.4 Strukturiertes Vorgehen bei Zielobjekten mit besonderen betrieblichen Anforderungen

Einen bewährten Ansatz stellt die Kombination aus standardisierter Risikoanalyse und -behandlung sowie bedarfsgerechter Erstellung zusätzlicher Maßnahmen bzw. Bausteine dar.

Wenn bei einem betrachteten Zielobjekt besondere betriebliche Anforderungen bestehen, ist das Zusammenspiel von technischen, infrastrukturellen, personellen und organisatorischen IT-Sicherheitsmaßnahmen neu zu bewerten. Diese Bewertung wird im Rahmen einer standardisierten Risikoanalyse dokumentiert und kann bei Betrachtung von gleichartigen Zielobjekten oder Szenarien wiederverwendet werden.

Führt das Ergebnis der Risikoanalyse dazu, dass die Gefährdungen nicht ausreichend durch die vorliegende Kombination von IT-Sicherheitsmaßnahmen abgedeckt sind, müssen zusätzliche Maßnahmen bzw. Bausteine erstellt werden. Gleiches gilt für Zielobjekte, bei denen besondere betriebliche Anforderungen dazu führen, dass sie nicht mit Standard-Bausteinen des IT-Grundschutzes modelliert werden können.

3.5 Managementkonzept für zusätzliche IT-Grundschutz-Bausteine

Die Entwicklung von zusätzlichen IT-Grundschutz-Bausteinen ist oft der einzige Weg, um spezifische Systeme des Informationsverbunds in der Modellierung abzubilden und relevante IT-Sicherheitsmaßnahmen zu identifizieren. Um den dabei entstehenden Aufwand nachhaltig zu reduzieren, hat sich die frühzeitige Entwicklung und Anwendung eines Managementkonzepts für die eigene Institution bewährt. Ein solches Konzept sollte folgende Aspekte betrachten und entsprechende verbindliche Vorgaben für die Umsetzung innerhalb der IT-Sicherheitskonzeption der Institution vorgeben:

- Zu berücksichtigende Vorschriftenlage
- Begründende Ausgangslage für die Modifikation bestehender Bausteine oder die Neuentwicklung von Bausteinen
- Vorgaben zur Anwendung den IT-Grundschutz-Katalogen vorliegender IT-Sicherheitsmaßnahmen
- Begründende Ausgangslage für die Modifikation oder Neuentwicklung von IT-Sicherheitsmaßnahmen
- Prüfverfahren und Qualitätssicherung
- Richtlinien zur Dokumentation und Veröffentlichung neuer Bausteine innerhalb der eigenen Institution
- Pflege und kontinuierliche Verbesserung der Bausteine über den gesamten Lebenszyklus

Diese Aufzählung deckt die notwendigen Mindestinhalte eines Managementkonzeptes zur Entwicklung von Bausteinen ab und ist institutionsspezifisch anzupassen.

4 Zusammenfassung und Ausblick

Um den Herausforderungen bei der zeitgemäßen Anwendung des IT-Grundschutzes in großen Institutionen zu begegnen, hat sich eine Vielzahl unterschiedlicher Lösungsansätze bewährt. Oft ist es nicht nur eine, sondern eine Kombination mehrerer Herausforderungen, die durch das ISMS einer großen Institution bewältigt werden muss. Daher ist eine enge Verzahnung dieser Lösungsansätze Grundlage für effizientes IT-Sicherheitsmanagement.

Die Abbildung des IT-Grundschutzes auf weitere Frameworks und Modelle kann eine solide Grundlage bilden, um die IT-Grundschutz-Methodik in die komplexen Prozesse und Betrachtungsweisen einer großen Institution einzubinden. Hierauf aufbauend fördern modulare Ansätze die Effizienz und Flexibilität der IT-Sicherheitskonzeption und die kontinuierliche Aufrechterhaltung des IT-Sicherheitsprozesses. Sind lange Projektlaufzeiten abzusehen, insbesondere bei der Planung und Realisierung neuer Systemanteile, sind Maßnahmen zum Erhalt von Wissen bei beteiligtem Personal und vertraglicher Anpassungsbedarf bei Änderungen in der Vorschriftenlage frühzeitig einzuplanen. Werden Zielobjekte mit besonderen betrieblichen Anforderungen identifiziert, bietet eine wiederverwendbare Risikoanalyse die Möglichkeit, Besonderheiten des Systems gezielt zu betrachten und effizient in eine modulare IT-Sicherheitskonzeption zu überführen. Besteht die Notwendigkeit für einzelne Zielobjekte zusätzliche Bausteine zu entwickeln, sollte das entsprechende Vorgehen einem Managementkonzept folgen, um Redundanzen zu vermeiden und Wiederverwendbarkeit zu optimieren. Ein solches Managementkonzept kann dann in den übergreifenden Anteil der IT-Sicherheitskonzeption der Institution aufgenommen werden.

In Zukunft wird der modulare Ansatz durch die Modernisierung des IT-Grundschutzes, insbesondere durch die Auswahl der Vorgehensweise, noch weiter unterstützt. Der modernisierte IT-Grundschutz wird verschiedene Methodiken anbieten, die sich an unterschiedliche Anwendergruppen richten und unterschiedliche Ziele verfolgen. „Beispielsweise kann sich eine Institution als Ziel setzen, zunächst möglichst flächendeckend alle Basis-Anforderungen umzusetzen ("Basisabsicherung"), um schnellstmöglich die größten Risiken zu senken, bevor die tatsächlichen Sicherheitsanforderungen im Detail analysiert werden. Ein anderer denkbarer Ansatz ist, sich zunächst auf den Schutz der essenziellen Werte der Institution („Kernabsicherung“) zu konzentrieren“ [MOD GS]. Für die IT-Sicherheitskonzepterstellung in komplexen Institutionen bietet sich somit eine Mischform aus diesen neuen Vorgehensweisen an. Die institutionsweite Basisabsicherung wird durch das übergreifende IT-Sicherheitskonzept gewährleistet und die verfahrensbezogenen oder systembezogenen IT-Sicherheitskonzepte stellen die Kernabsicherung des Informationsverbundes sicher.

Einen weiteren Vorteil für die IT-Sicherheitskonzeption in komplexen Institutionen bietet die Modernisierung des IT-Grundschutzes im Bereich der IT-Grundschutz-Profile. „Die modernisierten IT-Grundschutz-Profile sind als Schablonen zu verstehen, mit

denen verschiedene Anwender(-gruppen) selbstständig den IT-Grundschutz auf ihre Bedürfnisse anpassen können“ [MOD GS]. Hier ist denkbar, dass komplexe Behörden oder Unternehmen zentral definieren, welche Profile in welchem Teil des Informationsverbundes bzw. Modul der IT-Sicherheitskonzeption zur Anwendung kommen.

Literaturverzeichnis

[100-2] BSI-Standard 100-2. IT-Grundschutz-Vorgehensweise, Version 2.0. BSI, 2008.

[BDSG] Bundesdatenschutzgesetz. Juris, 25.2.2015.

[COBIT] COBIT 5, Framework for IT Governance, Risk, Security and Auditing. ISACA, 2013.

[EU Rl] EU Richtlinie 2002/58/EG (Datenschutzrichtlinie für elektronische Kommunikation). 2002.

[GSH] Geheimschutzhandbuch. 08.09.2014, BMWi.

[GSK] IT-Grundschutz-Kataloge des BSI, 14. Ergänzungslieferung. BSI, 2014.

[IETF] B. Khasnabish; e.a. IETF Cloud Reference Framework. 2011.

[ISO31000] ISO/IEC 31000 Standard, Risk Management. International Organization for Standardization, 2009.

[ITIL] IT Infrastructure Library. Axelos, 2013.

[MAIZ] Dr. Thomas de Maizière. Keynote zum BSI-Kongress 2015. www.bmi.bund.de/SharedDocs/Reden/DE/2015/05/ministerrede-bsi-kongress.html. BMI, 2015.

[MaRisk] Mindestanforderungen an das Risikomanagement, Rundschreiben 10/2012 (BA). BaFin, 2012.

[MOD GS] www.bsi.bund.de/DE/Themen/ITGrundschutz/IT-Grundschutz-Modernisierung/Kernpunkte/itgrundschutz_kernpunkte_node.html. BSI, 2015

[SOA] Ali Arsanjani; e.a. The SOA Manifesto. soa-manifesto.org/. 2013.

[VSA] Allgemeine Verwaltungsvorschrift des Bundesministeriums des Innern zum materiellen und organisatorischen Schutz von Verschlusssachen (Verschlusssachenanweisung). GBMI, 26.04.2010.

[ZDv] Zentrale Dienstvorschrift (ZDv) 54/100, IT-Sicherheit in der Bundeswehr. Bundeswehr, April 2013.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 171

Practitioners Track: Lessons learned bei KMUs aus dem Befall durch Ransomware

Florian Hertle[1]

Abstract: Am Beispiel eines konkreten Projekts wird gezeigt, welche aktuelle Gefährdungslage für KMUs aus grassierender Ransomware besteht. Mit Hilfe eines halb-automatisierten Vorgehens wurde bei einem KMU die IT-Landschaft, mit ca. 350 Clients und ca. 120 Servern, auf mit Ransomware infizierte Systeme geprüft. Per Anmeldeskript wurden von den Systemen verschiedene „Cryptowall 3.0"-spezifische Indikatoren gesammelt und anschließend per Skript aufbereitet. Auffälligkeiten wurden an den Systemen manuell überprüft. Zur Prävention vor neuen Malwareinfizierungen wurden Maßnahmen zur Erreichung eines Mindestniveaus der Informationssicherheit erarbeitet. Auf Basis eines Quick-Checks der vorhandenen technischen und organisatorischen Maßnahmen, wurde, per Bottom-Up Ansatz, entsprechende Quick-Wins abgeleitet.

Keywords: IT-Forensik, Verteidigungstechniken, Malware, Ransomware, Cryptowall 3.0, Mindestniveau, KMU, it.sec

1 Einleitung

Die Bedrohung durch Schadsoftware, die Dateien verschlüsselt und anschließend ein Lösegeld fordert (sogenannte Ransomware), hat sich im Jahr 2014 im Vergleich zum Jahr 2013 auf das 45-Fache erhöht [Wo15]. Auch im Jahr 2015 waren eine Vielzahl von Unternehmen von Cryptowall, Cryptolocker, Chimera und Co. betroffen. Ransomware beschränkt sich dabei nicht nur auf das Verschlüsseln lokaler Dateien, es werden auch Dateien auf Netzlaufwerken verschlüsselt, sofern der Benutzer darauf Zugriff hat. In der Regel beträgt das Lösegeld meist zwischen 300 und 1.000 EUR. Dieses Lösegeld ist mittels Bitcoins zu bezahlen.

Ransomware wird über verschiedenste Wege verbreitet. Der Hauptverbreitungsweg stellen E-Mails mit schadhaften Anhängen dar [Wo15]. Dabei kommen hauptsächlich ausführbare Dateien (.exe), JavaScript-Dateien (.js) und Word-Dokumente mit Makros (.doc, .docm) zum Einsatz. Die Polizei in Niedersachsen warnte am 12. Oktober 2015 von einer Ransomware namens „Chimera", die sich über fingierte Bewerbungen verbreitet hat. Die als Bewerbungsunterlagen getarnte Schadsoftware wurde dabei per Dropbox Link zur Verfügung gestellt [Po15]. Weitere Infektionswege stellen der Download von infizierten Dateien aus dem Internet und der Aufruf von

[1] it.sec GmbH & Co. KG, Einsteinstraße 55, 89077 Ulm, fhertle@it-sec.de

kompromittierten Webseiten dar. Seit Ende 2015 wird Ransomware vermehrt auch über Exploit Kits (Tool zur automatisierten Ausnutzung verschiedener Softwareschwachstellen) verteilt [Th15].

2 Aufgabenstellung

Bei einem KMU (kleine und mittlere Unternehmen) wurden drei Systeme im Zeitraum von wenigen Monaten mit der Ransomware „Cryptowall 3.0“ infiziert und sowohl lokale Dateien, als auch Netzlaufwerke verschlüsselt. Betroffen waren ein Geschäftsführer, der Leiter Vertrieb und ein Vertriebsmitarbeiter. In zwei der drei Fälle konnten die verschlüsselten Daten durch interne Sicherungsmaßnahmen größtenteils zurückgespielt werden. Im dritten Fall musste das Lösegeld bezahlt werden, um die Daten wiederherzustellen.

Aufgrund des Netzwerkdesigns wurde die Schadsoftware erst aktiv, als sich die Systeme mit einem externen Netzwerk (z.B. WLAN Hotspot) verbunden haben. Im internen Netzwerk wird der Port 80 an der Firewall geblockt, da die Internetverbindung nur über einen Proxyserver möglich ist. Somit bestand das Risiko, dass auf weiteren Systemen eine Malwareinfektion vorliegt/schlummert, ohne dass diese Kontakt mit dem C&C Server (Command & Control) aufnehmen konnte. Daraufhin wurde die it.sec beauftragt, die Frage zu beantworten, ob weitere Systeme mit der Ransomware „Cryptowall 3.0“ infiziert sind.

2.1 Informationen zum Unternehmen

Das Produktionsunternehmen ist in den letzten Jahren stark gewachsen, ohne dass die IT-Abteilung in gleichem Maße mit gewachsen ist. Derzeit betreuen fünf IT-Mitarbeiter eine sehr heterogene Systemlandschaft (ca. 120 Server und ca. 350 Clients) in mehreren Ländern. Als Betriebssystem kommen fast alle Versionen zwischen Windows 2000 Server bis Windows 7 zum Einsatz. Eine Standardisierung der Clients war bisher nicht möglich, da eine Vielzahl der Benutzer lokale Administratorberechtigungen für die Nutzung des ERP-Systems benötigen. Regelungen zum Umgang mit IT-Equipment waren nur rudimentär vorhanden.

3 Lösungsstrategie / Projektablauf

Die Fragestellung des Projekts lautete: „Sind weitere Systeme mit „Cryptowall 3.0“ infiziert?“. Aufgrund der Tatsache, dass aktuelle Malware nicht oder nicht sehr zuverlässig von Virenscannern erkannt wird, kann diese Frage nicht einfach durch einen Virenscan aller Computer beantwortet werden. Beispielsweise wurden durch die Cyber Threat Alliance im Zeitraum vom 01. August 2015 bis zum 31. Oktober 2015 1924

unterschiedliche „Cryptowall 3.0“ Varianten registriert, so dass für jede einzelne Variante ein spezielles Pattern des Virenscanners vorhanden sein müsste, um diese zuverlässig zu erkennen [Ct16]. Des Weiteren stellt eine manuelle Analyse aller Systeme eine nicht wirtschaftliche Lösung dar, weshalb es notwendig war ein Vorgehen zu entwickeln, um die Anzahl der manuell zu überprüfenden Systeme zu reduzieren. Hierzu wurde ein zwei-stufiges Vorgehen gewählt. In der ersten Stufe wurde ein Quick-Check der Informationssicherheit (Bestandsaufnahme der vorhandenen technischen und organisatorischen Maßnahmen) und eine Log-File Analyse der Firewall und des Virenscanners durchgeführt. Aufbauend auf die Ergebnisse der 1. Stufe wurde in der zweiten Stufe die Prüfung auf Malwareinfizierung durchgeführt. Hierzu wurde ein Skript entwickelt, um Indikatoren für eine „Cryptowall 3.0“ Infizierung zu sammeln, so dass auffällige Systeme identifiziert werden können, um diese manuell zu überprüfen.

Ausgehend von verschiedenen Malwareanalysen von „Cryptowall 3.0“-Binaries wurden folgende Indikatoren abgeprüft:

- Registry Einträge für Autostart
- Status und Einstellungen zu Volume Shadow Copy Service (VSS), da dies durch die Malware gestoppt wird
- Status zu verschiedenen Diensten (Sicherheitscenter, Windows Update, …), da diese ebenfalls durch die Malware gestoppt werden
- Windows Startkonfiguration
- Liste der Prozesse
- Liste der Dienste

Aufgrund der heterogenen Systemlandschaft sollten dabei hauptsächlich Windows eigene Tools zum Einsatz kommen, die in möglichst allen Betriebssystemversionen vorhanden sind, so dass hauptsächlich die Kommandozeilenbefehle „wmic“, „vssadmin“, „sc“ und „reg“ genutzt wurden. Das Batch-Skript konnte bei den meisten Systemen per Windows Gruppenrichtlinie als Anmeldeskript hinterlegt werden. Nach Ablauf einer gewissen Zeit wurden die gesammelten Indikatoren per Python Skript aufbereitet und die Auffälligkeiten (z.B. Autostart Eintrag in Appdata-Verzeichnis) ausgewertet. Dabei konnte ein System identifiziert werden, auf dem ein Malware Dropper (kompromittiert das System und lädt eigentlichen Schadcode nach) gefunden werden konnte. In der ersten Analyse wurden ca. 300 Systeme analysiert. Die Analyse wurde nach Ablauf einer weiteren Frist erneut durchgeführt, um die restlichen Systeme zu prüfen.

Im Rahmen der Log-File Analyse wurden bereits einzelne Systeme als auffällig identifiziert (geblockter Zugriff über Port 80 (http) und 443 (https) auf dubiose IP-Adresse). Da der Wissenstransfer zum Kunden elementarer Bestandteil des Projekts war, wurden die ersten Untersuchungen der auffälligen Systeme gemeinsam durchgeführt. Weitere Systeme konnten anschließend durch den Kunden selbst analysiert werden.

Im Rahmen des Lessons Learned Prozesses wurden Maßnahmen abgeleitet, um ein Informationssicherheitsmindestniveau zu erreichen. Hierzu wurde das Ergebnis des Quick-Checks mit der Risikosituation des Unternehmens abgeglichen. Per Bottom-Up Ansatz wurden Maßnahmen ausgewählt, die zur schnellen Verbesserung der Informationssicherheit beitragen.

4 Herausforderungen bei der operativen Durchführung

Durch die Entscheidung für einen halb-automatisierten Ansatz haben sich verschiedene Herausforderungen ergeben. Das Skript zum Sammeln der Indikatoren wurde von ca. 10% der Benutzer abgebrochen, so dass die Indikatoren nicht von allen Systemen vollständig vorlagen. Dies war möglich, da das Anmeldeskript nicht im Hintergrund ausgeführt wird. Um im Auswerteskript Fehler aufgrund von fehlenden Indikatoren zu vermeiden, wurde ein zusätzliches Prüfskript erstellt, welches eine Liste der Systeme mit vollständigen Indikatoren erstellt. Diese Liste wurde als Basis für das Skript zum Aufbereiten der Indikatoren verwendet.

Um sicherstellen zu können, dass alle Systeme analysiert wurden, ist eine Liste aller Systeme notwendig. Diese war jedoch nicht standardmäßig im Unternehmen vorhanden. Mit Hilfe der Softwareverteilungssoftware konnte eine Behelfsliste nachträglich erzeugt werden.

Die heterogene Systemlandschaft konnte auf Systeme mit mindestens Windows XP reduziert werden, so dass lediglich zwei unterschiedliche Skripte erstellt werden mussten: Ein Skript für Windows XP und Windows 2003 Server Systeme und ein zweites Skript für Versionen Windows 7 und die entsprechende Server Systeme.

Die älteren Systeme (ca. 5) wurden gegen Ende des Projekts manuell durch den Kunden überprüft.

Wie bereits erwähnt, arbeitet die Vielzahl der Benutzer mit lokalen Administratorberechtigungen. Diese werden von den Mitarbeitern genutzt, um beliebige Software zu installieren. Dies führt zu einem dazu, dass es nicht zielführend ist, die Systeme gegen ein „Baseline System“ abzugleichen und zum anderen hat es zu mehreren False-Positives geführt, da sogenannte PUPs (potentiell unerwünschte Programme) installiert waren.

Beim Auswerten der Indikatoren wurden die verschiedenen Indikatoren in einzelnen Text-Dateien (%COMPUTERNAME%-INDIKATOR-XX.txt) gespeichert. Teilweise waren diese bei unterschiedlichen Systemen unterschiedlich kodiert. Deshalb musste die Kodierung entsprechend in den Skripten hinterlegt und angepasst werden.

5 Fazit

Die Implementierung eines Virenscanners stellt keinen ausreichenden Schutz vor Schadsoftware dar. Weder die Ransomware, noch der Dropper wurde vom Virenscanner erkannt. Eine ganzheitliche Betrachtung der Informationssicherheit findet leider nur bei wenigen KMUs statt. Die derzeit akute Bedrohungslage durch Ransomware für KMUs sollte als Ausgangspunkt verwendet werden, die Informationssicherheit des eigenen Unternehmens zu überprüfen. Im vorgestellten Projekt wurde dies erst nach mehreren Vorfällen als notwendig erachtet. Deshalb beschränkte sich das Projekt nicht nur auf die Erkennung von weiteren Malwareinfizierungen, sondern der Kunde wurde auch bei der Verbesserung von präventiven Maßnahmen unterstützt und begleitet (z.B. Einführung einer ordnungsgemäßen Passwortpolicy und Vermeidung von lokalen Administratorberechtigungen). Der gewählte Bottom-Up Ansatz hilft, hier schnell Ergebnisse zu liefern, jedoch sollte langfristig auf einen Top-Down Ansatzes (beispielsweise in Anlehnung an die ISO/IEC 27001:2013) umgestellt werden. Auch ohne akuten Malwarevorfall sollte jedes Unternehmen überprüfen, ob alle Basisschutzmaßnahmen entsprechend implementiert sind und alle notwendigen Unternehmensbereiche abdecken.

Ergänzend zu den technischen und organisatorischen Maßnahmen muss die Awareness der Mitarbeiter zwingend verbessert werden. In den letzten Jahren hat die Qualität von SPAM und Phishing E-Mails zugenommen, so dass auch die Mitarbeiter besser auf die neuen Gefahren vorbereitet werden müssen. Einfache Qualitätskriterien, wie z.B. Rechtschreibung, Grammatik, Sprache, reichen zunehmend nicht mehr aus, solche E-Mails von regulären E-Mails zu unterscheiden. Vielmehr muss eine inhaltliche Prüfung der E-Mail auf Konsistenz (z.B. erwarte ich von einer Bewerbung eines Dachdeckers die Vorlage von verschiedenen Zertifikaten) und eine Prüfung der Dateiendungen der E-Mail Anhänge stattfinden.

Generell gilt, dass das Bezahlen der Lösegeldforderung vermieden werden sollte, um den Anreiz bei Angreifern nicht weiter zu erhöhen.

Literaturverzeichnis

[Ct16] Cyber Threat Alliance, http://cyberthreatalliance.org/cryptowall-dashboard.html, Stand: 27.01.2016.

[Po15] Polizei Niedersachsen, http://www.polizei-praevention.de/aktuelles/chimera-ransomware.html, Stand: 16.12.2015.

[Th15] Threatpost, Kaspersky Lab, https://threatpost.com/angler-exploit-kit-spreading-cryptowall-4-0-ransomware/115538/, Stand: 16.12.2015.

[Wo15] Wood, P.: Symantec Internet Security Threat Report, Volume 20, Mountain View, CA: Symantec Corp, 2015.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 177

Partitioners Track: Generating Security Vulnerabilities in Source Code

Felix Schuckert[1]

Abstract: This paper describes a framework, which modifies existing source code to generate security issues. An example plugin for generating SQL injection in Java source code is described. The generation process is based on static code analysis techniques like dataflow analysis and abstract syntax trees. The framework is evaluated with the help of Java projects from GitHub. One modified project was successfully used in a capture the flag event as a challenge.

Keywords: Software security, sql injection, generating, static code analysis

1 Introduction

Modern software products usually have a connection to the internet. All of them are exposed to attackers that want to exploit the software. The top security issues remained the same over the last years. Software developers have to protect their software from attackers by developing secure software. As stated in the paper [PE09], software security can't be provided only by the operating system. The software must have no vulnerabilities as well. Developers usually write tests, if common security vulnerabilities in their programs do exist, but this will not guarantee the security of the developed software. Instead it will verify the defined function requirements [Ba]. Static code analyzer tools help to find existing security issues in source code. These issues have to be reviewed and fixed manually by software developers. Fixing such issues requires a high amount of skill in software security. Another problem is that some security issues might not be detected by using code analyzer. At best all software developers should have software security skills to prevent software security issues preemptively in the development process.

How can developers gain software security skills? There are existing projects that provide manually created exercises which contain security issues. These exercises can be used to learn how the security issue can be exploited and how it can be fixed. The projects usually provide only a few exercises per security issue category. Having a framework that can be used for generating security issues in existing source code helps to generate a huge amount of exercises. The modified source code can be used to find issues in projects that contain more than a few lines of code, just created for the exercise. The modified project may be utilized to exploit and fix the issue as well.

Currently there are many papers about evaluating static code analyzers for detecting bugs and security issues. Some evaluated existing tools to detect security issues against SAMATE [SA10] database [GD13]. Also non-commercial static analysis tools like Yasca and

[1] HTWG Konstanz, Informatik, Brauneggerstr. 55, 78462 Konstanz, felix.schuckert@htwg-konstanz.de

FindBugs were evaluated for detecting security issues. [HHA14] Another research compared the detection of SQL injection vulnerabilities using static code analyzer and penetration testing. Static code analyzers found the vulnerabilities with a higher success-rate, but also have a higher false positive rate [AV09]. The evaluation of using static code analyzer for security verification was researched on existing tools like MOPS, Splint et cetera [ZSR14]. The usage of static code analyzers for detecting security issues is a common way of preventing vulnerabilities. Using static code analyzer for generating security vulnerabilities is the main goal of this work. There already exists a research about transforming *Statement* calls in Java to *PreparedStatement* calls using the refactoring features from Eclipse [AFS11]. This paper introduces a framework which is used for generating security vulnerabilities in existing source code.
Section 2 explains how the framework was implemented to generate security vulnerabilities. The next section 3 describes how the generation process is done using a SQL injection. The last section 4 shows how the modified source code looks like and points out the usage of the modified source code in a capture the flag event.

2 Framework for Generating Security Vulnerabilities

The framework implementation is method based. It will only generate security issues inside a method. The current implementation is focused on Java source code. The generation process requires basically three steps. The framework is using a plugin system for adding new generation processes. Each step can be implemented by plugins. In figure 1 the corresponding components are shown. The first step is marking methods with summaries. As described in [BC07] adding summaries to methods allows to scan in a wider scope than only scanning methods. Summaries will be added to method. These summaries can be included in the analysis of other methods, where corresponding methods will be called. For example the SQL injection plugin detects sanitize methods, which will be disabled in the generation process. The second step is the identification step, which searches for points of interests (POI). These will be used for generating the security vulnerability. As seen in the figure, the first two steps are using PMD [PM]. This tool is using abstract syntax trees (AST) for analyzing and provides features like dataflow analysis, XPath queries et cetera [Co05]. The last step is using the POIs found by the identifier and is modifying the AST to create the security vulnerability. This step is not done by using PMD. Instead it is using the tool Spoon [Sp]. This tool is using abstract syntax trees as well, but has the ability to modify it. Also the tool will generate the corresponding source code. It uses a simplified AST, which allows easier analyzing and modifying [Pa15]. The framework, explained in this paper, provides the functionality to find AST nodes in the Spoon AST from PMD nodes. This is basically done by the source code position. A plugin which implements these components will be able to generate security vulnerabilities in source code.

3 Plugin: SQL Injection

Evaluation of OWASP top ten [OW15] did show that injection is a top security issue category, which can be generated in Java source code. SQL injection was selected as the first

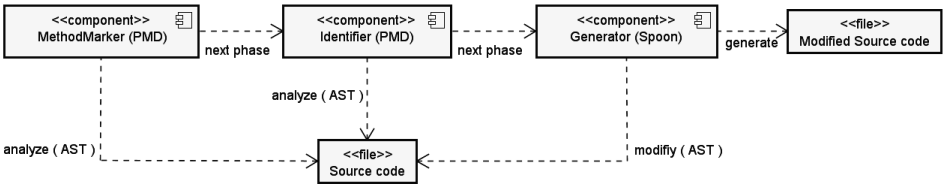


Fig. 1: Framework components in generation process.

plugin implementation for the framework because of the relevance and ease of implementation. This is used as a representative for the injection category. Each plugin requires a transformation model, which describes what source code parts have to be found and modified to generate the vulnerability. Figure 2 shows the transformation model for generating a SQL injection issue. The generation is similar to the detection of such an issue in source code. An examination of the Lapse+ project [OW] did show three basic parts of an injection vulnerability. Tainted sources which usually will be data from the user. These require no transformation to create the vulnerability. But they are important for checking, if any sanitize method will be used on them. Sanitize is the second part, which will remove or check for any suspicious characters. These methods have to be made useless so that the generated vulnerability can be exploited. Useless means that the function call still exists, but will have no effect and the program flow will still be the same as before. The last part is the sink. These are methods, where unchecked inputs should not be reached. The plugin will search for *PreparedStatements* instead, because they will be transformed to *Statements*. The corresponding method calls for setting the data and the SQL query statement have to be found as well. These parts are found by using PMD.

Tainted data sources have to be analyzed if they reach sanitize methods and sinks. This requires a modified tainted analysis instead of the usual tainted analysis described in [BC07]. It uses the dataflow analysis provided by PMD to detect sanitize method calls on tainted data. These calls will be saved and modified later in the generation step to be useless. The modified data flow analysis will check if tainted data will reach any set method call from the *PreparedStatement*. This will guarantee that the generated SQL injection can be exploited. The generator module implementation will use the found data and replace the *execute...()* call with the corresponding call using a *Statement*. The String concatenation will use the found SQL query statement and the set method calls. Exchanging the *execute...()* call will not change the behavior of the program in normal usage. It will only change the behavior, if the vulnerability will be exploited.

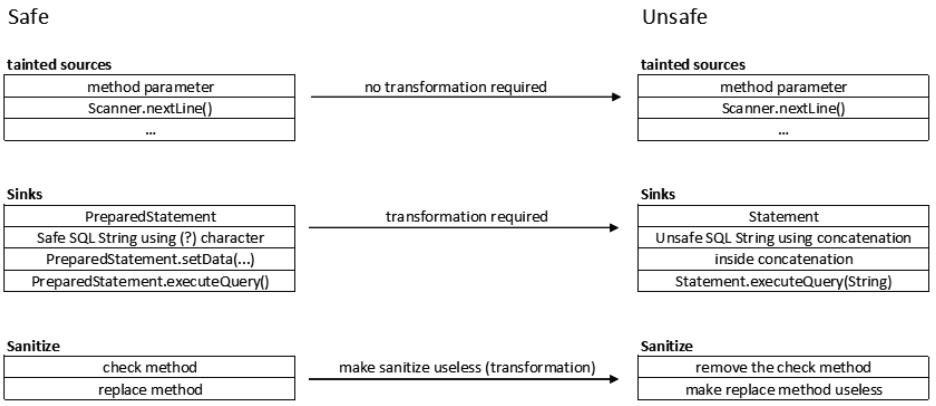


Fig. 2: Transformation model for creating a SQL injection vulnerability.

4 Evaluation and Usage

The generation process was evaluated using existing projects from GitHub. Following example shows an unmodified code snippet:

```
Connection connect = DriverManager.getConnection(
    "jdbc:mysql://127.0.0.1/test","","");
PreparedStatement ps = connect.prepareStatement("insert into teams
    (tname) values (?)");
ps.setString(1,tname);
ps.executeUpdate();
```

The plugin generated following source code by replacing the execute method call.

```
Connection connect = java.sql.DriverManager.getConnection(
    "jdbc:mysql://127.0.0.1:3306/test ?allowMultiQueries=true", "", "");
PreparedStatement ps = connect.prepareStatement("insert into teams
    (tname) values (?)");
ps.setString(1, tname);
String qryStr = "insert into teams (tname) values ('" + tname + "' )";
Statement statement=connect.createStatement();
statement.executeUpdate(qryStr);
```

The plugin also added the parameter for allowing multiple queries. This will encourage the developers, who are learning the SQL injection vulnerability, by having more possibilities to exploit the issue. Using multiple queries allows to add SQL commands like dropping the database or changing the admin password. The modified source code was evaluated using FindBugs [Fi] and Codepro AnalytiX [Co]. Both tools found the generated security vulnerability. Training the skills of developers to prevent SQL injection issues are still important because the issues found by tools have to be fixed manually.

The UCSB iCTF Competition [iC] provides a capture the flag event each year. These are attack/defense events where every team will maintain a server with running services. Each of these services have security vulnerabilities, which have to be fixed on the own server and have to be exploited on other servers to receive points. In 2015 each participating team had to submit one service with a security vulnerability. A project from GitHub modified by this framework was used for one service. This shows the first real usage of the framework. The event had 35 teams/services. The generated and submitted service was the fourth exploited in the contest. The submitted service was the second most exploited service from different teams. Stolen flags from the service were just at the 17th place. The main reason for this was, that some teams did find an exploit which caused a denial of service. This denial did also deny other teams from scoring points by using their exploits. The teams had to find and fix the SQL injection issue and additionally had to fix the database to get the service up on their own servers. This shows how important the prevention of SQL injection issues are. The framework allowed to generate exercises without any hassle using existing projects from GitHub.

5 Conclusion

Developers have to know security issues that have to be prevented during development. The developed framework allows to generate exercises, which can be used to teach developers how security issues can be exploited and how to prevent them. It can create different scenarios by using different source code. The issue will remain the same, but it will teach the developers to find the generated security issues in modified real life projects.

A basic framework for generating SQL injection was developed successfully. It uses abstract syntax trees for the analysis and generation parts. Dataflow analysis are used to do the modified tainted analysis. This allows to verify, if the vulnerability can be exploited. First usage in a CTF event showcased that the generated issues are easy to exploit. It requires further research to generate vulnerabilities with different complexity factors. This can be accomplished by having different permutations of the generation process.

The evaluation requires further steps. It has to be used as exercises for developers and the learning effect has to be examined. Further research in scanning other files, than source code like configuration files would allow to generate security vulnerabilities like cross-site-scripting. Also having the ability to generate vulnerabilities in memory unsafe languages as C/C++ would allow to generate buffer-overflow issues. As stated in the paper [EDB06] memory safe languages like Java do not allow to write memory parts which are not allocated for the corresponding variable. This will prevent any buffer overflows inside a Java program. This research did show an example generating SQL injections in the source code. Similar analyzing methods can be used in C/C++ because PMD supports it too. The generation part requires another tool for modifying C/C++ source code.

Literatur

[AFS11] Abadi, Aharon; Feldman, Yishai A.; Shomrat, Mati: Code-motion for API Migration: Fixing SQL Injection Vulnerabilities in Java. In: Proceedings of the 4th Workshop on Refactoring Tools. WRT '11, ACM, New York, NY, USA, S. 1–7, 2011.

[AV09] Antunes, Nuno; Vieira, Marco: Comparing the effectiveness of penetration testing and static code analysis on the detection of sql injection vulnerabilities in web services. In: Dependable Computing, 2009. PRDC'09. 15th IEEE Pacific Rim International Symposium on. S. 301–306, 2009.

[Ba] Baca, Dejan: Automated static code analysis: A tool for early vulnerability detection. ISBN 978-91-7295-161-7.

[BC07] Brian Chess, Jacob West: Secure Programming with Static Analysis. 2007. ISBN 978-0321424778.

[Co] CodePro AnalytiX. https://developers.google.com/java-dev-tools/codepro/, last visit: 2015-09-29.

[Co05] Copeland, Tom: PMD Applied. 2005. ISBN 0-9762214-1-1.

[EDB06] Emery D. Berger, Benjamin G. Zorn: DieHard: probabilistic memory safety for unsafe languages. ACM SIGPLAN Conference on Programming Language Design and Implementation, S. 158 – 168, 2006. ISBN 1-59593-320-4.

[Fi] FindBugs SourceForge. http://findbugs.sourceforge.net, last visit: 2015-09-29.

[GD13] Garbiel Diaz, Juan Ramon Bermejo: Static analysis of source code security: Assessment of tools against SAMATE tests. Information and Software Technology, 55:1462–1476, 2013.

[HHA14] Hamda Hasan AlBreiki, Qusay H. Mahmoud: Evaluation of static analysis tools for software security. Innovations in Information Technology (INNOVATIONS), 2014 10th International Conference on, S. 93 – 98, note = ISBN 978–1–4799–7210–4, 2014.

[iC] iCTF. https://ictf.cs.ucsb.edu/, last visit: 2015-12-12.

[OW] OWASP Lapse+. https://lapse-plus.googlecode.com/files/LapsePlusTutorial.pdf, last visit: 2015-09-29.

[OW15] OWASP Top Tent. https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project, last visit: 2015-09-03.

[Pa15] Pawlak, Renaud; Monperrus, Martin; Petitprez, Nicolas; Noguera, Carlos; Seinturier, Lionel: Spoon: A Library for Implementing Analyses and Transformations of Java Source Code. Software: Practice and Experience, S. na, 2015.

[PE09] Pistoia, Marco; Erlingsson, Úlfar: Programming Languages and Program Analysis for Security: A Three-year Retrospective. SIGPLAN Not., 43(12):32–39, Februar 2009.

[PM] PMD GitHub. https://pmd.github.io/, last visit: 2015-09-03.

[SA10] SAMATE - Software Assurance Metrics and Tool Evaluation. https://samate.nist.gov, last visit: 2015-12-13.

[Sp] Spoon. http://spoon.gforge.inria.fr/, last visit: 2015-10-02.

[ZSR14] Zhioua, Zeineb; Short, Stuart; Roudier, Yves: Static Code Analysis for Software Security Verification: Problems and Approaches. In: Computer Software and Applications Conference Workshops (COMPSACW), 2014 IEEE 38th International. S. 102–109, 2014.

Doktoranden-Forum „Sicherheit 2016“

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 185

Sicherheit im Softwareentwurf – Integrierte Modellierung von Security und Softwarearchitektur

Stefanie Jasser[1]

Abstract: Die Anforderungen an die Informationssicherheit heute eingesetzter Softwaresysteme steigen zunehmend. Dennoch werden sie während der Softwareentwicklung meist vernachlässigt. Softwarearchitekturen beeinflussen die Erfüllung nichtfunktionaler Eigenschaften wie Informationssicherheit wesentlich. Existierende Ansätze erlauben keine explizite Modellierung der Informationssicherheit eines Softwaresystems auf der Abstraktionsebene der Softwarearchitektur.
Der zu erarbeitende Ansatz ermöglicht die integrierte Modellierung von Aspekten der Informationssicherheit und den Architektureigenschaften eines Softwaresystems. Es ist geplant, die Evaluation dieses Ansatzes im Rahmen einer Feldstudie durchzuführen, d. h. die Anwendbarkeit wird durch den Einsatz in realen Softwareprojekten gezeigt.

Keywords: Secure Software, Security Engineering, Software Engineering, Softwarearchitektur, Modellierung, Informationssicherheit

1 Motivation

Softwaresysteme sind ein wesentlicher Bestandteil der heutigen Gesellschaft und Industrie. Viele dieser Systeme sind kritisch, d. h. sie sind etwa notwendig für die Fähigkeit eines Unternehmens, seine Dienstleistungen oder Produkte anzubieten. [MMR10] Zugleich werden Softwaresysteme offener und leichter angreifbar: Die Anforderungen an die Sicherheit dieser Softwaresysteme steigen. Dabei werden Anforderungen an den Schutz gegen Angriffe (Security) und den Schutz gegen unbeabsichtigte Ereignisse (Safety) unterschieden.

Nach [BSYJ15, SC09] muss die Sicherheit bereits während der initialen Entwicklung eines Softwaresystems berücksichtigt werden, um diese Anforderungen gerecht erfüllen zu können. Das bedeutet, Sicherheitsziele sind bereits in der Anforderungsanalyse zu erheben und zu formulieren und sollten den Softwareentwurf beeinflussen. Dies gilt besonders für den Entwurf der Softwarearchitektur, die für die Erfüllung nichtfunktionaler Eigenschaften eines Softwaresystems wesentlich ist. Zu diesen Eigenschaften zählen auch Security und Safety. Grundlegende Sicherheitsmängel (Security bzw. Safety Flaws) sind schon in der Softwarearchitektur erkennbar. Sie sind in späteren Entwicklungsphasen nur noch schwer zu beheben [FKK+14, SCH04] und müssen daher frühzeitig korrigiert werden. In den anschließenden Phasen Implementierung und Softwaretest muss die Einhaltung der Sicherheitsziele und der Softwarearchitektur sichergestellt und überprüft werden. Analysewerkzeuge auf Codeebene unterstützen die Durchführung von Penetrationstests oder Reviews.

[1] Universität Hamburg, Fachbereich Informatik, Arbeitsbereiche Sicherheit in verteilten Systemen und Softwareentwicklungs- und Konstruktionsmethoden, Vogt-Kölln-Straße 30, 22527 Hamburg, jasser@informatik.uni-hamburg.de

[NNY10] geben einen Überblick über Arbeiten, die die Erhebung und Formulierung von Sicherheitsanforderungen behandeln. Weniger Autoren befassen sich mit der Berücksichtigung von Sicherheitsaspekten im Softwareentwurf. Einige Ansätze unterstützen den Entwurf fehlertoleranter Softwaresysteme und beziehen Safety-Aspekte ein. [WK04, VKE12] Die meisten Ansätze, die Security im Softwareentwurf berücksichtigen, behandeln nur einzelne Security-Aspekte. Alle Arten von Security-Aspekten eines Softwaresystems können in keinem Ansatz modelliert werden. [BSYJ15]

Auch nach der initialen Entwicklung verändern sich die Umgebung und die Anforderungen an ein Softwaresystem. Es muss an diese neu entstehenden Anforderungen angepasst werden. Dieser Prozess heißt Softwareevolution und kann auch die Softwarearchitektur betreffen (Architekturevolution). [SB12, MMR10] Durch Änderungen des Systems und seiner Umgebung können wiederum Bedrohungen und Security Flaws entstehen. Forschungsarbeiten zur Berücksichtigung der Security während der Evolution eines Softwaresystems fehlen weitgehend. [NNY10] Dies gilt insbesondere für die Architekturevolution. [FKK+14, Ren06]

2 Stand der Forschung

Die Mehrheit der existierenden Ansätze erlauben nur die Modellierung ausgewählter Sicherheitsaspekte. Meist ist dies die Zugriffskontrolle (Access Control). Als Basissprache nutzen viele Autoren die *Unified Modeling Language* (UML) [BSYJ15]: Ein Ansatz, der die Modellierung verschiedener Sicherheitsaspekte erlaubt, ist UMLsec. UMLsec nutzt den vorgesehenen Erweiterungsmechanismus von UML, indem es ein Profil definiert. [Jür02, JW02] UML-Diagramme können so durch sicherheitsspezifische Stereotypen und Tags annotiert werden. Die meisten dieser Sicherheitsaspekte zählen zur Safety und die Modellierung erfolgt nicht explizit auf der Architekturebene. Dies gilt auch für andere auf UML basierende Ansätze wie SecureUML, das die Modellierung von rollenbasierten Zugriffskontrollen ermöglicht. [LBD02, BSYJ15, BDL06].

Mit MASC (Modelling Architectural Security Concerns) erweitern [SYB+15] UML, um Architekturentscheidungen für ausgewählte Security-Konzepte wie Kryptographie oder Interception zu dokumentieren. Eine automatisierte Validierung der Modelle findet nicht statt. MASC wurde bisher nicht anhand realer Projekte evaluiert.

Für die Erweiterung von UML um Sicherheitskonzepte spricht, dass UML als de-facto Standard der Industrie gilt, um Modelle in der Softwareentwicklung zu erstellen. Softwareentwickler und -architekten sind den Umgang mit UML-Modellen gewohnt. Die Notation wird in der Industrie jedoch häufig nur informell genutzt, da formal korrekte UML-Modelle leicht sehr komplex werden. Diese Komplexität wird durch Erweiterungen wie zusätzliche Elemente oder Annotationen noch erhöht, [SYB+15] sodass eine Nutzung außerhalb des wissenschaftlichen Kontextes unwahrscheinlich ist. Dies gilt insbesondere für eine automatisierte Validierung, die eine formal korrekte Modellierung voraussetzt. Meist werden UML-Diagramme nicht für die Modellierung auf Architekturebene genutzt.

Auch andere Architekturbeschreibungssprachen (Architecture Description Languages, ADLs) wurden um Sicherheitsaspekte erweitert. Diese Erweiterungen beziehen sich wiederum nur auf einzelne Security-Aspekte und erlauben deren ganzheitliche Modellierung nicht. Der in [Ren06, RTDR05] vorgestellte Ansatz Secure xADL beschreibt beispielsweise Aspekte der Zugriffskontrolle auf Architekturebene.

Das *Threat Modeling* ist ein Mittel zur Analyse von Bedrohungen für ein Softwaresystem. Eine verbreitete Threat Modeling Technik ist der iterativ angelegte STRIDE-Ansatz von [HL06], dessen Name ein Akronym der sechs Bedrohungskategorien: Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege. Er wird im Microsoft Secure Development Lifecycle eingesetzt, der Bedrohungsanalysen anhand eines Datenfluss-Diagramms vorsieht. Um identifizierte Bedrohungen zu mindern, werden Gegenmaßnahmen in Form von Änderungen des Softwaresystems ergriffen. Das neue Modell kann wiederum auf Bedrohungen analysiert werden. Weder diese Bedrohungen noch die ergriffenen Gegenmaßnahmen sind explizit im Datenfluss-Diagramm sichtbar und ein direkter Abgleich mit üblichen Softwarearchitektur-Modellen ist schwierig: Die initiale und evolutionäre Entwicklung des Softwaresystems wird unzureichend unterstützt.
Bedrohungen können auch mithilfe anderer Techniken modelliert werden. Eine solche Technik sind die von [Sch99] vorgestellten Attack Trees, deren Wurzelknoten ein Angriffsziel beschreiben. Die Kindknoten stellen Möglichkeiten dar, dieses Angriffsziel zu erreichen. [SO05] nutzen Misuse Cases oder Abuse Cases zur Analyse von Bedrohungen während der Anforderungsanalyse. Sie modellieren angelehnt an UML Use Case-Diagramme missbräuchliche Verwendungen eines Softwaresystems. Diese Techniken zur Bedrohungsanalyse beschreiben die Nutzung separater Modelle, die während der Anforderungsanalyse oder des Softwareentwurfs erstellt werden. Sie erlauben keine integrierte Modellierung und Validierung der Sicherheitsziele und Architektureigenschaften eines Softwaresystems.

Zusammenfassend ist eine umfassende, leichtgewichtige Notation für die Modellierung von Security-Aspekten im Softwareentwurf notwendig. Insbesondere fehlt die Möglichkeit zur integrierten Modellierung von Security-Aspekten und der Softwarearchitektur. Dies stellen auch andere Autoren fest: [BSYJ15, FKK+14, NNY10].

3 Forschungsfragen und methodisches Vorgehen

Um Entwickler und Architekten in der Entwicklung sicherer Softwaresysteme zu unterstützen, muss in der Entwurfsphase des Softwareentwicklungsprozesses angesetzt und die Modelle bis zur Abschaltung des Systems aktuell gehalten werden. Im Promotionsvorhaben sollen daher folgende Forschungsfragen bearbeitet werden:

Wie lassen sich Security-Aspekte und Softwarearchitektur integriert modellieren?

Es wird eine integrierte Modellierung der Softwarearchitektur und ihrer Eigenschaften angestrebt. Die erarbeitete Notation soll eine leichtgewichtig Modellierung der Security-Aspekte eines Softwaresystems ermöglichen. Je höher die Zahl der benötigten Modelle

ist, desto aufwendiger wird deren Synchronisation. Diese Herausforderung tritt besonders während der Evolution der Modelle im Rahmen der Softwareevolution auf. Werden Security-Aspekte und Architektureigenschaften eines Softwaresystems in wenigen Modellen dargestellt, kann dies eine erhöhte Modellkomplexität verursachen. Zwischen diesen Herausforderungen muss eine Abwägung getroffen werden. Die Erfüllung modellierter Security-Ziele auf Architekturebene sowie deren Konsistenz sind im angestrebten Konzept automatisiert validierbar. Im Rahmen dieser Validierung soll auch untersucht werden, inwiefern Softwareentwickler und -architekten ohne Expertenwissen in Security im Entwurf sicherer Systeme unterstützt werden können.

Welche Wechselwirkungen bestehen zwischen Security-Zielen und Architektureigenschaften im Rahmen der Evolution?

Es sollen die Auswirkung veränderter Security-Anforderungen auf die Softwarearchitektur und vice versa untersucht werden. Basierend auf den Ergebnissen, ist ein Konzept zu erarbeiten, das die Evolution der integrierten Modelle unterstützt. Dies kann durch eine Entscheidungsunterstützung realisiert werden, die Eigenschaften und Entscheidungshistorie des Legacy-Systems einbezieht. Für Modelländerungen ist zu untersuchen, wie eine erneute Validierung effizient gestaltet werden kann.

Wie erodieren Security-Funktionen in Softwaresystemen?

Während der Entwicklung kommt es zu Abweichungen zwischen der geplanten Soll-Architektur und der tatsächlich implementierten Ist-Architektur eines Softwaresystems. Es soll untersucht werden, inwieweit sich die Erosion von Security-Funktionen von der Erosion anderer Software-Funktionen unterscheidet. Darauf aufbauend werden Gegenmaßnahmen wie Security-Refactorings erarbeitet.

Die Basis des Promotionsvorhabens bildet die Analyse existierender Arbeiten zur Modellierung und Dokumentation von Softwarearchitekturen und Security-Aspekten sowie anerkannter Security-Lösungen auf Architekturebene. Solche Lösungen können Security-Muster, -Taktiken oder Best Practices aus Wissenschaft und Industrie sein. Auf den Ergebnissen aufbauend, wird ein Konzept erarbeitet, das die leichtgewichtige Modellierung der Security-Aspekte eines Softwaresystems erlaubt. Die Anwendbarkeit dieses Konzepts wird durch Feldstudien [Pre01] in realen Softwareprojekten evaluiert.
Zur Evolution und Erosion von Security-Funktionen in Softwarearchitekturen existieren nur wenige Arbeiten. [FKK+14] Daher werden die Auswirkungen evolutionärer Änderungen mittels einer Diskussion und anhand realer Softwareprojekte untersucht. Dies umfasst Änderungen der Security-Ziele und der Softwarearchitektur. Die Betrachtung der Erosion der Security-Funktionen erfolgt im Rahmen dieser Untersuchung. Die Erkenntnisse werden genutzt, um das Modellierungskonzept um Mechanismen zu erweitern, die evolutionäre Änderungen und Verbesserungen unterstützen.

4 Zusammenfassung

Die Anforderungen an die Security von Softwaresystemen steigen kontinuierlich. Existierende Ansätze ermöglichen es nur, ausgewählte Security-Aspekte im Softwareentwicklungsprozess zu berücksichtigen. Für die Erfüllung der Anforderungen ist ihre frühzeitige Einbeziehung jedoch notwendig. Das Promotionsvorhaben strebt die Integration von Security-Aspekten und der Softwarearchitektur in der Entwurfsphase an. Für die tatsächliche Anwendbarkeit ist ein leichtgewichtiges Konzept wesentlich, das die Modellierung aller benötigten Informationen erlaubt. Dieses Modellierungskonzept wird im Kontext des Softwarelebenszyklus entwickelt: Im ersten Schritt wird die initiale Entwicklung eines Softwaresystems betrachtet, in der die Softwarearchitektur erstmalig entworfen wird. Im zweiten Schritt werden evolutionäre Änderungen der Softwarearchitektur und der Security-Ziele betrachtet. Da Softwaresysteme während ihrer Entwicklung erodieren, erfolgt eine Untersuchung der Erosion von Security-Funktionen und möglicher Gegenmaßnahmen. Die Ergebnisse werden argumentativ und in realen Softwareprojekten evaluiert.

Literaturverzeichnis

[BDL06] David Basin, Jürgen Doser und Torsten Lodderstedt. Model Driven Security: From UML Models to Access Control Infrastructures. *ACM Transactions on Software Engineering and Methodology*, 15(1):39–91, 2006.

[BSYJ15] Alexander van den Berghe, Riccardo Scandariato, Koen Yskout und Wouter Joosen. Design notations for secure software: A systematic literature review. *Software & Systems Modeling*, 2015.

[FKK+14] Michael Felderer, Basel Katt, Philipp Kalb, Jan Jürjens, Martín Ochoa, Federica Paci, Minh Sang Le Tran, Thein Than Tun, Koen Yskout, Riccardo Scandariato, Frank Piessens, Dries Vanoverberghe, Elizabeta Fourneret, Matthias Gander, Bjørnar Solhaug und Ruth Breu. Evolution of Security Engineering Artifacts: A State of the Art Survey. *International Journal of Secure Software Engineering*, 5(4):48–98, 2014.

[HL06] Michael Howard und Steve Lipner. *The security development lifecycle: SDL, a process for developing demonstrably more secure software*. Microsoft secure software development series. Microsoft Press, Redmond, Washington, 2006.

[Jür02] Jan Jürjens. *Principles for Secure Systems Design*. Dissertation, Oxford University, University of Oxford, 2002.

[JW02] Jan Jürjens und Guido Wimmel. *Security Modelling for Electronic Commerce: The Common Electronic Purse Specifications*, Jgg. 74 of *IFIP International Federation for Information Processing*, Seiten 489–505. Kluwer Academic Publishers, Boston, 2002.

[LBD02] Torsten Lodderstedt, David Basin und Jürgen Doser. *SecureUML: A UML-Based Modeling Language for Model-Driven Security*, Jgg. 2460 of *Lecture Notes in Computer Science*, Seiten 426–441. Springer, Berlin, Heidelberg, 2002.

[MMR10] Tom Mens, Jeff Magee und Bernhard Rumpe. Evolving Software Architecture Descriptions of Critical Systems. *Computer*, 43(5):42–48, 2010.

[NNY10] Armstrong Nhlabatsi, Bashar Nuseibeh und Yijun Yu. Security Requirements Engineering for Evolving Software Systems: A Survey. *IJSSE*, 1(1):54–73, 2010.

[Pre01] Lutz Prechelt. *Kontrollierte Experimente in der Softwaretechnik: Potentiale und Methodik*. Springer, Berlin Heidelberg, 1. Auflage, 2001.

[Ren06] Jie Ren. *A Connector-Centric Approach to Architectural Access Control*. Dissertation, California State University, Long Beach, California, USA, 2006.

[RTDR05] Jie Ren, Richard N. Taylor, Paul Dourish und David F. Redmiles. Towards An Architectural Treatment of Software Security: A Connector-Centric Approach. *ACM SIGSOFT Software Engineering Notes*, 30(4):1–7, 2005.

[SB12] Lakshitha de Silva und Dharini Balasubramaniam. Controlling software architecture erosion: A survey. *Journal of Systems and Software*, 85(1):132–151, 2012.

[SC09] Sarah Spiekermann und Lorrie Faith Cranor. Engineering Privacy. *IEEE Transactions on Software Engineering*, 35(1):67–82, 2009.

[Sch99] Bruce Schneier. Attack Trees: Modeling Security Threats, 1999.

[SCH04] Adam Sachitano, Richard O. Chapman und John A. Hamilton. Security in Software Architecture: A Case Study. In *from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004*, Seiten 370–376, 2004.

[SO05] Guttorm Sindre und Andreas L. Opdahl. Eliciting security requirements with misuse cases. *Requirements Engineering*, 10(1):34–44, 2005.

[SYB+15] Laurens Sion, Koen Yskout, Alexander van den Berghe, Riccardo Scandariato und Wouter Joosen. MASC: Modelling Architectural Security Concerns. In *2015 IEEE/ACM 7th International Workshop on Modeling in Software Engineering (MiSE)*, Seiten 36–41, 2015.

[VKE12] Timo Vepsalainen, Seppo Kuikka und Veli-Pekka Eloranta. Software architecture knowledge management for safety systems. In *2012 IEEE 17th Conference on Emerging Technologies & Factory Automation (ETFA 2012)*, Seiten 1–8, 2012.

[WK04] Weihang Wu und T. Kelly. Safety tactics for software architecture design. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004*, Seiten 368–375, 2004.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 191

A Mechanism Design for Privacy-Preserving Computation on Shared Data

Saffija Kasem-Madani[1]

Abstract: In times of surveillance and data retention, sharing information often comes together with privacy concerns. However, information sharing has benefits, e.g. sharing log files for including the knowledge gained from a broader view for security analysis, or sharing healthcare data for the use in studies for improving medical treatments. We present an information sharing framework design that meets both privacy and utility requirements of the information sharing parties. We utilize homomorphic encryption and show how it can be used for offline data analysis.

Keywords: encrypted computing, homomorphic encryption, multiparty computation, information sharing, privacy, utility.

1 Introduction

When information is shared among multiple parties, the interests of the data holders in keeping cleartexts confidential for privacy collide with the interests of the data receiver in maximal utility.
Consider the following application scenario: Data holders want to share log files (e.g., see [SY04]) with a centralized analysis entity, called data analyzer. The data holders are interested in the results of the analysis the analysis entity performs. The analysis entity profits from collecting log files from multiple sources because it gives him a broader view on the field, e.g. a business area.
On the other hand, data holders want to keep their data confidential. One reason is the privacy of the data owners. This contradicts the data analyzer's interest in data utility.
In this work, we assume that the data holders and the data analyzer formulated a compromise between the need of privacy and utility, i.e. a privacy-utility trade-off, in contracts called policies. A policy consists of conditions the exchanged data has to fulfill in order to meet all utility options that meet both the analyzer's and the holders' requirements. Utility options formulated in a policy may include the ability to calculate overall sums of numerical data for statistical analysis, or disclosing IP addresses in a log file under certain conditions, or checking equality of encrypted data. The conditions stated in a policy must be selected very carefully to ensure that the privacy of the data owners stays preserved. For a formulation of such policies, we refer to [KMM15].
Before sharing the data with the analyzer, the data holder must transform the data such that it will meet the policy. We call the result of such a transformation *data appearance*.
Depending on the use case and the sensitivity of the data under consideration, the selection

[1] Rheinische Friedrich-Wilhelms-Universität Bonn, Institut für Informatik, Friedrich-Ebert-Allee 144, 53113 Bonn, kasem@cs.uni-bonn.de

of the provided utility still must keep so many information confidential that an attacker is not be able to use the utility properties of the data appearances for information linkage and correlation attacks with the help of external information.
One approach that can be used is homomorphic encryption. In homomorphic encryption, the cleartext data p_1, p_2 are encrypted with an encryption mechanism E to $E(p_1), E(p_2)$. E ensures that certain operations can be performed on $E(p_1), E(p_2)$ and are equivalent to operating on the plaintexts p_1 and p_2. I.e., $E(p_1) * E(p_2) = E(p_1 + p_2)$ for appropriate operations $*$ and $+$. Note that the result of the encrypted computation is encrypted.
In the application scenario described above, the data holder would use a homomorphic encryption mechanism to encrypt the cleartext content of the log files to data appearances. After receiving the data appearances, the analyzer performs the agreed operations on the data for analysis. To check the operation results, the analyzer needs to decrypt them. This usually requires knowledge of the secret (private) key. However, having access to the private key enables the data analyzer to decrypt the content of the data appearance and hence, violating the privacy of the data owners.
To overcome the described problem, we present a mechanism that ensures that only results of allowed computations can be disclosed (i.e. decrypted) to the analyzer. We describe the parties of the mechanism and the design assumptions made for ensuring the privacy preservation of the shared data. The mechanism utilizes Paillier, a public-key probabilistic homomorphic encryption scheme [Pa99]. With the prospected upcoming improvement of homomorphic encryption schemes[3], we aim at providing feasible solution for its use for privacy preserving information sharing.

2 Description of the Mechanism

We describe the stakeholder of the mechanism, how they receive the data and metadata, and privacy-preserving decryption of results of operations performed on the Paillier encrypted data.

2.1 Stakeholders

For privacy-preserving information sharing for data analysis, we consider a mechanism that consists of the data holders, the data analyzer, the activator, the private-key holder, the preprocessor, and the decryptor. The stakeholders ensure that no single party gains more information than stated in the privacy-utility policy. The stakeholders of the mechanism are described as follows:

- The **data holder** has full access to plaintext data. His purpose is to share the data with the analyzer. He creates data appearances of Paillier-encrypted data. There can be multiple data holders.
- The **data analyzer** receives the data appearances from the data holders. He performs operations over the encrypted data contained in the collected data appearances.

[3] See e.g. the HEAT project https://heat-project.eu/

- The **preprocessor** holds the public keys necessary for completing the operations instead of the analyzer. This limits the types of operations the analyzer can perform.
- The **activator** has the main part in ensuring that only analysis results are disclosed to the analyzer. For this purpose, he checks for the validity of the encrypted results against a database of all hash values of encrypted components of data appearances. If the results are valid, he activates the decryption process.
- The **private-key holder** holds all private keys of all data appearances for the decryption process. The private keys are bound to certain data rather than being bound to a subject.
- The **decryptor** decrypts the encrypted operation results and sends them to the analyzer.

2.2 Preparation and distribution of data

After stating the policy, i.e. agreeing on the utility that the content of the data appearances of a data holder has to fulfill, the data holder initiates the data sharing process. He sends the data appearance D and and identifier id to the analyzer, enabling it to perform analysis operations as stated in the policy. The identifier id is used later to identify the correct public and private key for the decryption steps. Depending on the utility conditions stated in the policy, it may necessary to use more than one key pair. In this case, the data holder defines an order of the keys and assigns each key pair one hash value as follows: the first key pair is assigned the hash value $h(id)$. The second pair is assigned the hashed hash $h(h(id))$, the third pair is assigned $h(h(h(id)))$, and so on. We refer to an $i-$th hash $h_i(h_{i-1}(\cdots h_2(h_1(id))))$ with $h_i(id)$.
After generating the data appearance according to the policy, the data holder sends it to the analyzer. The analyzer can perform analysis operations on the data appearances.
The data holder computes hashes $h(d)$ for each encrypted item in the data appearance D and a hash value of the appearance's identifier, $h(id)$. D contains m encrypted items. Then he sends the set of hashes $\{(h(id), h(d_i)|i \in \{1,...,m\})\}$ to the activator. If multiple key pairs are used, $h(id)$ is altered with the corresponding hash value.
Then, the data holder sends the set of all the private keys together with the corresponding hashed identifiers $\{(k_{priv,i}, hash_i(id))|i \geq 0\}$ to the key holder. The private-key holder identifies the correct keys and send them to the decryptor for decryption.
Then, the data holder sends all public keys together with the hashed identifiers to the preprocessor, as described for the private keys. I.e., the preprocessor receives a set $\{(k_{pub,i}, hash_i(id))|i \geq 0\}$. The preprocessor can now perform the preprocessing of the encrypted operation results received by the data analyzer.
Note that the keys are bound to certain data, not to subjects. If multiple data holders are involved in an information sharing action, they may agree on using the same keys for encrypting certain values for allowing the analysis of aggregated data from multiple sources.
After the data distribution, each of the stakeholders is prepared for taking part in the decryption process without further interaction with the data holder.

2.3 Data analysis

In this work, we are considering data that is encrypted with the Paillier cryptosystem. Being partially homomorphic, it is possible to perform homomorphic additions of ciphertexts. The public key is required for calculating the correct modulus of the added or multiplied ciphertexts. When the public key is given, it is possible to multiply and add a cleartext to an encrypted value. We omit the mathematical explanations here and refer to the original reference [Pa99].

2.4 Decryption of the operation results

The data analyzer perform the analysis operation op on some $d \subset D$, yielding $enc'(op(d))$, where $enc(op(d)) = enc'(op(d)) mod n^2$, n is one component of the public key. After performing the required operations on the encrypted data, the analyzer wants to know the cleartext values of the results for interpretation and further processing. In order to complete the calculation, he requests the preprocessor for applying the correct modulus calculation. For this, he sends a request that contains the partially calculated result and the hashed identifier for the public key $\{enc'(op(d)), h(id)\}$. With this, he initiates the conditional decryption process. I.e. the operation result will be decrypted if the data analyzer can show that he is not requesting for the decryption of data appearance content.
Once the preprocessor has received the request for preprocessing, he selects the correct public key, calculates the modulus and applies it to $enc'(op(d))$ with the result $enc(op(d))$. Then he requests the activator for decryption. For this, he sends a message that contains the encrypted result of the operation together with the hashed identifier of the key $\{enc(op(d)), h(id)\}$.
The activator checks whether the hashed value of $enc(op(d))$ is in the database. If this holds, he sends a "FIN" message to the preprocessor, denying the possibility of decryption. The preprocessor sends the "FIN" message to the analyzer.
If the activator finds no matching entry in the database, he assumes that $enc(op(d))$ is an encrypted operation result and not contained in the data appearance. This can be assumed due to the use of Paillier, a probabilistic encryption scheme.
The activator then creates randomly selected nonce n and sends it to the analyzer. The activator sends one message that contains $h(id)$ and the nonce n to the private-key holder, and one message $\{enc(op(d)), n\}$ to the decryptor.
On receiving $\{n, h(id)\}$, the private-key holder selects the private key that corresponds to $h(id)$ and sends it together with the nonce n to the decryptor. The decryptor receives the key, identifies it as belonging to $enc(op(d))$ using the nonce and decrypts the analysis result. The decryptor then sends the decrypted analysis result $op(d)$ together with the nonce n to the data analyzer.

2.5 Security assumptions and analysis

In the mechanism, we assume the honest-but-curious adversarial model [PMB]. I.e., we assume that the attacker is curious about knowing the cleartexts of messages she sees,

but follows defined protocols and would not break any security mechanisms. Further, we assume that the activator would not collaborate with the private-key holder to decrypt the data. Nor would he send the hashed identifiers to the decryptor to let him reconstruct which key has been used for a certain dataset. We also assume that the neither the activator nor the private-key holder would collaborate with the data analyzer to decrypt components of data appearances. As a result, the activator would neither learn the analysis results nor the cleartexts of the data appearances content. Due to the use of nonces that do not depend on the data indices, i.e. the hash values, for addressing the decryption part of the process, the decryptor would neither be able to learn to which dataset a decrypted analysis result belongs, nor would he be able to map an already received key to a future ciphertext for decryption. If he would store each private key he receives and collaborates with the data analyzer for disallowed decryption, he would be, at most, enabled to perform a brute-force attack on data items using all the keys.

We assume that the analyzer is not able to perform *misused operations* on the data. With misused operations, we mean adding constants homomorphically to the encrypted data. In Paillier cryptosystems, this requires access to the public key. Due to the probabilistic property of the Paillier cryptosystem, the activator would identify such forged ciphertexts as results of valid computation and, hence, activate the decryption process. Once the analyzer has received the decrypted result, he would get the cleartext data by subtracting the previously homomorphically added constant. In order to prevent the analyzer from performing misused operations, the public key is not made accessible to him. Instead, we introduced the preprocessor who holds the public keys and is assumed not to collaborate with the analyzer for misuse operations.

Assuming the honest-but-curious adversarial model in the current mechanism design may not be suitable for very sensitive use cases. To serve more stringent adversarial models, the tasks of the stakeholders may be redistributed based on secure multiparty computation schemes [Go98]. This would ensure that an attacker cannot successfully misuse the mechanism to gain disallowed information unless all parties actively collaborate with him.

3 Related Work

Homomorphic encryption schemes are a well-studied field [RSA78] [Pa99] [Ge09]. Performing operations on encrypted data for privacy preservation is an actual research field. Mechanisms for machine learning [GLN13] [Bo15] and recommending [Er12] on encrypted data have been presented. Mechanisms for secure multiparty computations [Go98] are widely used and have been adapted to threshold homomorphic encryption [CDN01].

4 Conclusions and Future Work

We presented the design of a mechanism for privacy-preserving decryption of data analysis results on homomorphically encrypted data. We described the stakeholders and their capabilities, explained the roles of each stakeholder, how he data and metadata is distributed and how the mechanism performs a privacy-preserving decryption. We performed a brief

security analysis. In order to weaken the trust assumptions made, new parties and a secret sharing mechanism [Sh79] may be used.
Implementing and evaluating the mechanism would show the real-world applicability of our mechanism. One may investigate the applicability of the mechanism for different homomorphic encryption mechanisms. The use of threshold homomorphic encryption [CDN01] may lead to a simplified mechanism for data analysis. However, the design of practicable threshold homomorphic encryption is still an open research problem.

References

[Bo15] Bost, Raphael; Popa, Raluca Ada; Tu, Stephen; Goldwasser, Shafi: Machine Learning Classification over Encrypted Data. In: 22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2014. 2015.

[CDN01] Cramer, Ronald; Damgård, Ivan; Nielsen, Jesper B.: Advances in Cryptology — EUROCRYPT 2001: International Conference on the Theory and Application of Cryptographic Techniques Innsbruck, Austria, May 6–10, 2001 Proceedings. Springer Berlin Heidelberg, Berlin, Heidelberg, chapter Multiparty Computation from Threshold Homomorphic Encryption, pp. 280–300, 2001.

[Er12] Erkin, Z.; Veugen, T.; Toft, T.; Lagendijk, R.L.: Generating Private Recommendations Efficiently Using Homomorphic Encryption and Data Packing. Information Forensics and Security, IEEE Transactions on, 7(3):1053–1066, June 2012.

[Ge09] Gentry, Craig et al.: Fully homomorphic encryption using ideal lattices. In: STOC. volume 9, pp. 169–178, 2009.

[GLN13] Graepel, Thore; Lauter, Kristin; Naehrig, Michael: ML confidential: Machine learning on encrypted data. In: Information Security and Cryptology–ICISC 2012, pp. 1–21. Springer, 2013.

[Go98] Goldreich, Oded: Secure multi-party computation. Manuscript. Preliminary version, 1998.

[KMM15] Kasem-Madani, Saffija; Meier, Michael: Definition of Availability Policies for Data Pseudonymization Using XACML. IFIP Summer School on Privacy and Identity Management, August 2015.

[Pa99] Paillier, Pascal: Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In (Stern, Jacques, ed.): Advances in Cryptology, EUROCRYPT 99, volume 1592 of Lecture Notes in Computer Science, pp. 223–238. Springer Berlin Heidelberg, 1999.

[PMB] Paverd, AJ; Martin, Andrew; Brown, Ian: Modelling and Automatically Analysing Privacy Properties for Honest-but-Curious Adversaries. Technical report.

[RSA78] Rivest, Ronald L; Shamir, Adi; Adleman, Len: A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM, 21(2):120–126, 1978.

[Sh79] Shamir, Adi: How to Share a Secret. Commun. ACM, 22(11):612–613, November 1979.

[SY04] Slagell, Adam J.; Yurcik, William: Sharing Computer Network Logs for Security and Privacy: A Motivation for New Methodologies of Anonymization. CoRR, cs.CR/0409005, 2004.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 197

Towards Adaptive Event Prioritization for Network Security - Ideas and Challenges

Leonard Renners[1]

Abstract: In the network security domain Intrusion detection systems (IDS) are known for their problems in creating huge amounts of data and especially false positives. Several approaches, originating in the machine learning domain, have been proposed for a better classification. However, threat prioritization has also shown, that a distinction in true and false positives is not always sufficient for a profound security analysis. We therefore propose an approach to combine several aspects from those two areas. On the one hand, threat and event prioritization approaches are rather static with fixed calculation rules, whereas rule learning in alert verification focuses mostly on a binary classification and does not target rule parameters. In this paper we highlight specifics and challenges in event prioritization rules and describe first ideas and challenges towards solutions by the means of automatically learning these aspects.

Keywords: Network Security, Intrusion Detection, Machine Learning, Event Priority

1 Introduction

A lot of effort has been spent in reducing the number of false positives in network security alerts. Especially IDS are known for their problems in creating huge amounts of data, and studies have shown that up to 99% of them belong to the group of false positives, meaning they either were not successful or generally not relevant[JD02]. Multiple approaches have been conducted to tackle these challenges. Mostly, additional information is used to confirm or contradict an alert. Some approaches suggest to utilize techniques from the machine learning domain and learn from labeled attacks and the analysts behavior.

Nevertheless, the amount of reports can still be huge, due to the fact that many alerts can be generated for a single attack and also non-important, but successful, incidents are also classified as true positives. On the other hand, not all true negatives should be discarded. The information gained, even from unsuccessful attacks, can be substantial in protecting the network and especially preventing future incidents. For example an attack, which has not been successful, but was conducted from an IP-address in the corporate network may very well be a security incident, although the alert itself was a false positive.
Promising results towards this problem have been shown by trying to evaluate more information and calculate event priorities, instead of solely classifying the alerts in true/false positives. One major problem still is the fact, that these rather complex rules have to be written by a domain expert and mostly consist of static calculation formulas or cumbersome property files. These approaches therefore require high configuration efforts and are limited in their applicability in dynamic environments, which networks typically are.

[1] Hochschule Hannover, Fakultät IV, Abteilung Informatik, Ricklinger Stadtweg 120, 30459 Hannover, leonard.renners@hs-hannover.de

We are arguing, that one possible solution is to specifically design rules to target prioritization parameters in order to enable an easier understanding and maintenance of such rules. Furthermore we want to try to adapt some of the learning processes from advanced IDS to learn these rules from rated events and related background knowledge. In this paper we identify specifics of prioritization focused rule based systems and discuss resulting challenges for an automated learning. Note that event prioritization and thus the proposed approach is well suited, but not limited, to the domain of IDS alerts.
The remainder of this paper is structured as follows. Section 2 covers important work in related research areas, including automatic IDS alert classification and threat prioritization. The subsequent section 3 discusses our ideas towards the specifics of rules for event prioritization and gives first thoughts on the learning of each aspect. The final section 4 gives concluding remarks and an outlook on the next steps and future lines of research.

2 Related Work

Alert correlation and alert verification are techniques to improve the precision in classifying IDS alerts in order to identify true positives and thereby reduce the amount of alerts generated. While alert correlation uses the information from different IDS sensors to rate alerts, alert verification is the process to include additional information, mostly about the underlying topology and especially the target, to determine the probability of a successful alert [Yu04][MHT05]. Further research went towards the automated learning of IDS rules with background information in order to generate automatic classifiers. Pietraszek et al. [Pi04] built an adaptable learner for alert classification - especially suitable for alert verification - using machine learning techniques.

More recently, some research has been conducted towards a more general rating of events instead of classifying true and false positives. An implementation framework approach on cyber threat prioritization has been presented by Townsend and McAllister [TM13] where they analyze the aspects that influence the priority of an event. They conclude on three major aspects - impact, likelihood and risk - and further describe each one of them. Kim et al. [Ki14] propose a similar approach in their ACCEPT framework - a threat prioritization framework for cyber network defense. They develop why and how incoming cyber events can be prioritized in order to make quick and effective decisions, but also give several concrete metrics and formulas for different aspects of a cyber event, e.g. the importance of the target host or connected entities, beside the severity of the attack itself.

The approach presented in this paper utilizes some of the related concepts, but extends the functionality, especially by learning the concepts in the background knowledge in order to identify important rule elements, instead of solely focusing on the outcome. Furthermore the nature of rules for IDS alert verification and event prioritization differ, as the more recent approaches have shown, and concepts for learning can not be adopted directly. Event and threat prioritization thus far relies on pre- and user-defined rules without the possibility to automatically learn or adapt these aspects and is also lacking an appropriate syntax dedicated to this problem area.

3 Adaptive Prioritization Rules: Specifics and Challenges

State-of-the-art rule languages and rule based systems are generally capable of event pattern evaluation and can also be used to prioritize the events. Meanwhile, it is important to keep in mind the need for understandable and modifiable security policies in order to not only identify and deal with the events, but to also make sure to include the security administrator and follow the company compliance guidelines.
In this section we identify some special properties of prioritization focused rules. These can be used to construct a concept to explicitly target these challenges and the resulting rules become much more intuitive to understand and write.
Figure 1 gives a definition of the structural elements of a rule and an example (in a pseudo-code event-condition-action syntax) to further explain the different aspects of a rule and their specifics for prioritization.

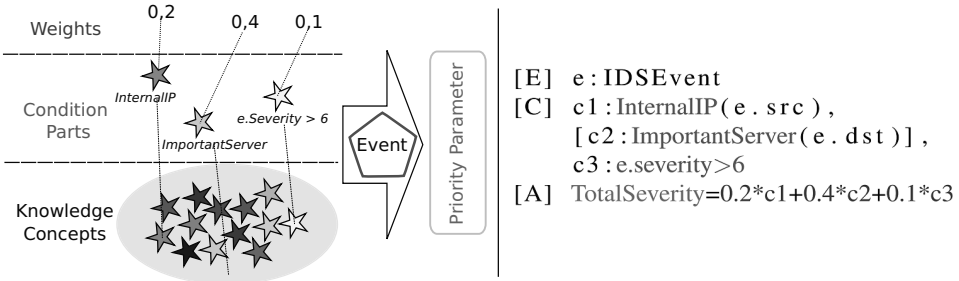


Figure 1: Structural Elements of a Rule

A rule for prioritization, in our understanding, can be divided in four constituents: *Knowledge concepts*, *condition parts*, *weights* and *priority parameters*.
Knowledge concepts comprise all available building blocks which can (potentially) be used as part of a rule condition. Condition parts then define the set of concepts which are used to describe one particular rule and form the influential factors on the focused priority parameter - the calculation result - of the event. They can be optional (square brackets) or necessary for a rule to fire.
Finally, weights are those factors which define the effect of each condition part on the priority parameter.

3.1 Knowledge Concepts

So most rule based system do not really target and highlight a defined set of possible concepts to use. Typically only attributes of the events are used in Boolean expressions, as in the example regarding the severity. For prioritization it is advantageous to define more abstract concepts to generalize from single attribute values. This makes the conditions reusable and thereby allows for an easier maintenance, since only the concept and not every single rule has to be adapted. Another aspect of the generalization is the encapsulation of more complex expressions. While a pattern concerning an integer is expressed quite easily inline, other conditions can be more complex. And often times not each individual attribute

value is important or crucial for decision making but its affiliation to a certain group of values. In the example the concrete ip-address is not really relevant, but its affiliation to the group *InternalIP* (the ip-addresses belonging to the internal corporate network). An inline expression would make the rule increasingly complex and harder to understand, especially for non-domain experts and analysts confronted with the rule.

Meanwhile, one can argue, that the introduction of knowledge concepts only shifts the complexity from the rules towards the background knowledge. While this is true, we believe that this shift also helps to cope with the complexity of the overall system since each part can now be tackled separately. Nevertheless, the definition of concepts and rules still is a complex problem and we believe that the user needs assistance in this area. So the task is to identify potentially relevant attributes and more importantly group and abstract from single values. Such learning can mostly be achieved unsupervised, e.g. by clustering techniques, but the following challenges have been identified and have to be overcome in future research:

Attribute Types: The value space of attributes can vary and need to be treated differently. E.g. numerical attributes could be separated into ranges whereas strings might be grouped with respect to their Levenshtein distances.

Indirect grouping: Some information might also not be relevant directly, but specific constellations of attribute manifestations. An example for a relevant relation is if the exploited vulnerability of a specific attack is actually present in a software running on the target system.

Evaluation: The quality assessment of different abstractions and grouping is difficult, but necessary to choose suitable concepts. The definition of a group or cluster does not always have a direct result, but can only be implicitly observed by its impact on the other aspects, especially the condition parts and finally the classification results.

3.2 Condition Parts

The concrete choice of elements out of the subset of knowledge concepts used to describe one specific rule are called condition parts. *Required* arguments specify the necessary parameter for the rating process and are the typical elements of rule based system. They can also describe a superimposed detection of a relevant situation (incident detection or alert correlation), and naturally influence the priority parameter.
Optional condition parts on the other hand are evaluated if all necessary requirements are fulfilled and used for a more accurate description and calculation of the resulting priority parameter. The example describes a rule to consider all rules that originate from the internal network and have a high base severity. But the fact, that the target is also important is an optional factor, that makes the alert even more relevant. To put it the other way around, we want to express patterns comparable to the following: an attack with a low severity is not relevant, even if it targets an important server, but the importance of the server is an influential factor for generally relevant attacks.

In order to automatically infer these parts typical rule induction techniques can serve as a basis for supervised learning. Although extensions are necessary in order to use the numerical priority parameters and not a static true/false classification. Furthermore we have to keep in mind the dependencies to the defined knowledge concepts. If the concepts are too specific, i.e. too many attribute characteristics exist, the condition parts also become increasingly specific and target only special cases, whereas the goal is to generalize rules and provide a generic rating, applicable for upcoming alerts.

3.3 Weights

Most use-cases for rule based systems define rules in order to detect relevant situations or abstract from a huge amount of low-level data. In these environments the rule parameters typically do not really influence the rule result, beside the fact they lead to a new (complex) event and some attributes may be adopted. In prioritization rules however, specifically with optional parameter, the condition parts need to explicitly influence the calculation as the rule result. In addition, rules and their results have to be comparable in order use the calculated rating to sort the events. The considered parameters need to have specific ranges in which all rules map their individual calculations, although the weights are specific for each rule. The same holds true for the relations between condition parts themselves. While most parts influence the value of a prioritization aspect directly, there might be some situations where all other factors become completely irrelevant. One example is the knowledge, that a target is not vulnerable against a certain attack (e.g. the software is known to be patched against this certain attack) and therefore the alert probability should always be zero.

The calculation of weights might be included in the rule induction process, e.g. by applying a linear regression for rule induction. Otherwise this challenge has to be tackled separately. The weights could be inferred from the already defined condition parts, when examining different calculation results in similar rules. Either way, it is an interesting question how the identified peculiarities regarding relativity and absolute values can be targeted.

4 Outlook and Future Work

In this paper, we presented our ideas towards adaptive event prioritization to tackle the challenge of the manual and complex rule design of threat prioritization systems. Therefore, we identified special requirements and challenges for a better rule design in order to provide a distribution of complexity and thereby more understandable rules. Further, we gave some remarks towards challenges in automatically learning these aspects in order to reduce the effort and assist the user. However, the proposed ideas are only first thoughts and further research is required in every single direction and especially in the combination of the subproblems.

For knowledge concepts, most importantly a way is needed to validate learned concepts, probably using the whole framework, but this involves the other learning techniques.
Condition parts should be analyzed with regard to the extracted patterns. For example

whether only complete patterns are valid rules or if sub-expressions can be considered. And further how they can be used to refine the concepts, e.g. if too complex rules emerge from the rule induction.
Weights can be learned from similar rules, but a concept has to be developed to include dependencies between parameters and rules.

Lastly, and parallel to the conceptual parts of the learning process, we also want to take a closer look at the syntactical representation. As highlighted in the paper, prioritization rules have some specifics which could best be represented by a syntax that explicitly tackles these requirements. The learning process could preferably generate such *complete* rules in a resulting framework.

One side issue occurring in the analysis of the proposed techniques is to acquire suitable data for testing purposes. This also concerns the task of an evaluation of the system. Labeled test data could be used to compare the overall result, e.g. the prediction accuracy. But further tests regarding the induction of the single rule aspects and especially towards the usability are desirable and need to be designed and conducted. Anyhow, not much data is publicly available in this area, especially rated alerts with background knowledge. So one auxiliary task will be to find or generate and prepare training data, i.e. complement existing data sets by suitable priority parameters.

References

[JD02] Julisch, Klaus; Dacier, Marc: Mining intrusion detection alarms for actionable knowledge. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 366–375, 2002.

[Ki14] Kim, Anya; Kang, Myong H.; Luo, Jim Z.; Velasquez, Alex: A Framework for Event Prioritization in Cyber Network Defense. Technical report, July 2014.

[MHT05] Mu, Chengpo; Huang, Houkuan; Tian, Shengfeng: Intrusion Detection Alert Verification Based on Multi-level Fuzzy Comprehensive Evaluation. In (Hao, Yue; Liu, Jiming; Wang, Yuping; Cheung, Yiu-ming; Yin, Hujun; Jiao, Licheng; Ma, Jianfeng; Jiao, Yong-Chang, eds): Computational Intelligence and Security, Lecture Notes in Computer Science 3801, pp. 9–16. Springer Berlin Heidelberg, December 2005.

[Pi04] Pietraszek, Tadeusz: Using Adaptive Alert Classification to Reduce False Positives in Intrusion Detection. In (Jonsson, Erland; Valdes, Alfonso; Almgren, Magnus, eds): Recent Advances in Intrusion Detection, Lecture Notes in Computer Science 3224, pp. 102–124. Springer Berlin Heidelberg, September 2004.

[TM13] Townsend, Troy; McAllister, Jay: , Implementation Framework â Cyber Threat Prioritization, September 2013. ref 28 from kim et al.

[Yu04] Yu, J.; Ramana Reddy, Y.V.; Selliah, S.; Kankanahalli, S.; Reddy, Sumitra; Bharadwaj, Vijayanand: TRINETR: an intrusion detection alert management systems. In: 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004. pp. 235–240, June 2004.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 203

Designing resilient and secure smart micro grids

Siavash Valipour[1]

Abstract: The research presented in this extended abstract paper depicts a smart grid management framework which enables a decentralized and autonomous organization of the energy participants within these grids. Based on basic requirements for operating such systems, challenges and tasks are being discussed here as a basis for future research. The discussion encompasses both fields of electrical engineering and computer science. The presented grid coordination and energy transfer schemes are briefly regarded and openly discussed.

Keywords: Smart Grid, Micro Grid, Energy Network, Decentralized Power Sources

1 Introduction

The supply of energy is on a transition from a traditional, centralized "powerplant-to-consumer" tenet to an increasingly decentralized approach in countries like Germany, respecting renewable energy policies. In recent years, smart grids have been discussed to empower an IT-supported "smart" supply system, where an intelligent system manages energy produced and consumed within the grid. The energy participants of such grids are usually proposed to be organized in decentralized clusters. The idea behind smart micro grids (from now on: "smart grids") is to produce and consume ideally renewable energy locally, i.e., near the source in order to decrease transportation losses and costs for infrastructure investments. Implementing such a grid management system requires knowledge from both electrical engineering and computer science and faces challenges such as grid reliability, security and resilience. The members of a micro grid consist of heterogeneous parties, that is, consumers, producers and storage systems which are equipped with smart-metering and control devices, providing bidirectional information and energy flows. Here, the classic household can act both as a consumer and as a producer at the same time, for example by having solar panels installed on the roof which supply parts of its immediate neighborhood. There exist many definitions and surveys depicting major differences between the smart grid and the traditional grid. A good insight view can be obtained in [Fa10], [Fa14], [HHM11], [Fa12b] and [Fa12a]. In this paper, we present our research on a formal, resilient and decentralized smart grid model. The overall goal of this proposed system consists of providing certain autarchy and self-management aspects in these dedicated networks. The term "resilience" aims here at a high degree of service availability, making the grid robust to failures and attacks. This presented research plan deals with selected integral topics related the smart grid's performance and stability:

[1] Technische Universität Darmstadt, TK, Hochschulstr. 10, 64289 Darmstadt, valipour@tk.tu-darmstadt.de

1. Model: Defining an applicable smart grid coordination model is needed. This particularly includes formal statements to form logical groups and subgroups within the decentralized network as well as defining goals.
2. Coordination: These groups need coordination, thus an underlying dynamic coordination model is required. This can be done by centralized/decentralized managers, called coordinators, who are elected by a leader election process.
3. Prioritization: In situations as in energy scarcity phases, a priority based energy distribution system seems favorable, e.g., classifying the needs of hospitals higher than households. We discuss such a proposed priority model.

2 Research Questions

Based on the smart grid model (SGM) and the available capabilities of the participants within the model, algorithms for energy management, emergency plans etc. can be defined.

2.1 The Model

In the literature, both Hashmi et al. and Fang et al. survey smart grid technologies and approaches in [HHM11] and [Fa12a], respectively. The German state funded project *Modellstadt Mannheim* [MV12] proposed a *cell* driven topology for the grid. As this model offers a rather fixed mapping of how devices and buildings are set up in this system, dynamic processes which arise during shortages are not being taken care of with the desired efficiency and thus leave room for improvement. Several non-exhaustive research tasks can be identified when designing our model: First, **maximize the overall availability of energy** to present consumers. Second, support island operation modes in a case of a (partly) blackout. Third, enable dynamic and demand-based allocation of energy. Fourth, support management and transfer of locally produced energy.

The future grid's desired ability to separate itself from the greater network (island operation) requires new concepts and investments in communication-enabled electrical equipment. From an IT-based point of view, these islands need to be operated and coordinated according to the up-to-date parameters, e.g. sensor data, and constraints, e.g. line capacities. Two concepts are favored here, the hierarchical and the autonomous approach. The centralized, hierarchical approach where a central control station controls its local domain grid, allows grid operators a total supervision over the entire grid, but also creates a single point of failure. Given that the future grid's stability will heavily depend on the reliability of the IT infrastructure, this approach should be questioned.

Otherwise, in an autonomous system, each part of the grid acts as self-regulatory systems where individual decisions and actions are based on locally available data and communication. The decentralized system takes electrical sensor data via communication as input in order to determine the current (self-)behavior of the local entity. Decisive units are presumed within the local grid need to retain its stability. This can be achieved by dynamic and adaptive promotion of normal grid stations ("buildings") into a decisive station through

leader election, which will be further described later on. Our current and future research deals with investigating a well suited decentralized smart grid coordination model, aimed to be self-regulatory and resilient in terms of service availability.

When applying such a systems-theoretic concept onto smart grids, new challenges arise as future work. We define this very formal model to incorporate all smart grid participants (electrical and information layer) into one framework. This encompasses buildings, devices, power stations, servers, lines and connections. The model strives to maintain an equilibrium between the produced and consumed amount of energy at all times as one of its goals. Available energy flexibilities on demand and producer side are considered as well as emergency situations and strategies. In cases of danger, e.g., parts of the grid being in exceptional states due to an electrical short circuit, detected malicious parts should be excluded from the larger entity so that a further instability of the grid is mitigated. The formal model is going to be designed for dealing with these issues by defining monitoring functions and polling sensors. The functions are evaluated regularly by the coordinator and a set of best practice rules and strategies, e.g. opening switches, throttling down consumers, activating generators etc. will be defined in order to mitigate potential risks and failures. For that, several constraints from the engineering perspective are not to be neglected: A "logically" separated building might still be physically connected to the grid and thus influence the area. Solutions for this matter might be remote controllable switches and "electricity gateways" installed at the house connector level. These gateway devices are envisioned as supply influx delimiters, installed in the house's basement and connected to the grid's communication network as well as the main supply line. If the gateway receives an order from a coordinating authority to limit its demand, it is assumed this device will do so by ensuring that consumers within that building are switched off or kept on standby until the total consumption is below the threshold. This allows the coordinator to actively conduct the grid operation given scenarios. The model's concept is to be proven by theorems while its abstracted feasibility is meant to be evaluated and researched on by simulation.

2.2 Leader Election

There are a wide variety of efficient leader algorithms (LEAs) mentioned in the literature. The bully algorithm described by Garcia-Molina in [GM82] along with its sped up versions by Lee et al. [LC02] as well as by Arghavani et al. [AAH11] are to name among others. The algorithm by Yu et al. [Yu09] describes a more electricity-grid-fitting approach where each system starts off by assessing a self-evaluating process based on criteria such as computing and communication capacities. Then these values are propagated and a vote for the leader is conducted. In these models, when the current leader fails, the approach is to simply conduct a new election, triggered by anyone noticing the failure. While the number of reelections is reduced in improved versions, the idea of a complete reelections appears to be excessive, as all other nodes except of the leader may still be online. Also, denial-of-service attackers easily could exploit this behavior. An apparent idea is the application of a "leader / deputy leader" system which could enable quick substitution in case of failure. As the coordinator is the key player in our grid model, it is essential to consider the resilient nature of our model also when designing a LEA. It is assumed, that a failing

coordinator triggers the participants into an emergency mode and limits consumption of energy to a minimum in order to – if possible – avoid major outages and instabilities until a new coordinator is found. An adversary could be interested in inserting unauthorized, or manipulating existing, machines in the network and try to promote himself as a new coordinator to conduct his malicious intents. He could also deliberately disturb the election process preventing mutual consent and keeping the grid in under-resourced exceptional state. The integrity of the LEA is therefore of utmost importance.

Our own, very preliminary and shortened approach focuses on a LEA based on a multi-step voting scheme. It is assumed, that the participants Π are able to communicate with each other. The crucial part is depicted as follows: First, each participant willing to participate in the election process will do an self-assessment regarding its own leader qualifications. The assessment algorithm might consider the machine's available computational resources, its communication capacities, its physical location in the grid, its machine type and so forth. All of these factors are weighted by the algorithm. The assessment values are then propagated to everyone in election. These received values are then multiplied by a trust factor $t_i \in [0..1]$, where each participant has stored such a dynamic factor for every other participant's unique ID $i \in \Pi$. The participant with the highest result is then openly voted by multicast. As the voting is open, the participants will accept the participant with the highest vote-count as the new leader.

Trust values are stored individually on non-volatile memory and set to a standard value initially, but change dynamically over periods of time, due to the participants' individual experience with that leader in the past. If, for example, a leader was noticed for keeping up the grid only for short periods of time, or if the supply under that coordinator was regularly sub-optimal, then the assigned trust value for that particular leader might decrease. We expect to come up with such tailored LEA. The method will be evaluated formally by assessing the number of messages sent in cases where a leader is needed or fails.

2.3 Priority Model

Available capacities and consumption-needs in the local grid are part of the energy balance which should remain equated. In times where the demand exceeds the supply, active energy management is proposed to keep the grid operational.

We propose a priority scheme which allows to decide on which buildings to supply first. For example, a hospital might appear more important to supply with scarce energy than a normal household. Horsmanheimo et al. show by a formal proof the theoretic property of such a priority grid model in [Ho14]. However, given that there is only limited energy available, one might argue that a coffee machine inside the hospital might appear less important to supply than a refrigerator inside a standard household. Therefore, we extend prevalent priority models from "building-only" over the domain of smart devices. We assume for this priority model, that devices are partitioned into device priority classes inside the smart home, according to their type. Furthermore, the coordinator knows all its connected consumers, their priorities and the consumer's aggregated energy-need bound to

device priority classes. It is also assumed that adversaries are able to change neither the building priority nor device priorities on their own nor replay requests to the coordinator in order to gain benefits or disturb the grid. The installed and operational energy gateway devices allow the coordinator to actively scale the energy influx into buildings.

The "importance" of incoming energy requests is obtained by adding (or multiplying) the device's priority (class) to the requesting building's priority. Therefore, the consumer's request consists of the device class and the energy amount. Now, energy is dealt out by the coordinator from the (in numbers) lowest priority class to the highest. We formally introduce thus a set of $B := \{1, ..., n\}$ different building priorities and $D := \{1, ..., m\}$ device classes and $m, n \in \mathbb{N}$. An example follows: the refrigerator (device priority $d_k \in D$) in a kiosk (building priority $b_k \in B$) is in the "total priority class" $p = d_k + b_k$. The coordinator would first try to serve in its micro grid all other priority requests smaller in numbers than p unless surplus energy is available for p.

Nevertheless, a few tasks persist: First, which entities and devices should be assigned priorities? It is expected that regulating authorities will have to decide on this. Second, seek methods to reduce priority propagation and processing complexity. Third, how much better is the priority system in scarcity operation mode? We expect to publish simulated results in near future by comparing our priority model to a non-prioritized / FIFO model.

3 First Results

We conducted a grid simulation to find potentials in decentralized and autonomous supply. In the traditional grid system, a failing transformer station would drive the subnet behind the transformer offline (blackout) even though local generators might be available and could potentially operate parts of the grid. Under given assumptions described in our paper currently under submission, two models of decentralized energy supply were evaluated in randomly generated, real-world based scenarios: The first model operates with respect to a fixed cellular model with low dynamics. Here, the island mode, i.e., a connected component of the smart grid graph, is operational when all provisioned participants behind a failing local substation transformer are technically available, can communicate and have available producers and consumers present. The improved model allows operation even when only sub-components of the grid are found to be available. Our simulation ran 10,000 passes with randomly generated sub-grids which are based on parameters obtained from a real-world site in a medium-sized German city. Our results show that the proposed dynamic island approach can supply an average fraction of 22.08% of the sub-grids when the substation transformer fails. This encourages deeper research in this field.

The proposed priority model was also tested in similar grid topologies with randomized priorities. The simulation showed the theoretical feasibility of our model, by correctly preferring priority consumers over others, while at the same time still respecting production and line capacities. As such a model appears to not be implemented elsewhere, it is to determine appropriate ranges of priorities for both buildings and devices in order to assert reasonable and just consumer partitioning during scarcity-plagued operations.

4 Conclusion

Smart grids are future models of electricity distribution empowering the renewable energy transition. We propose a model which facilitates a decentralized way of operation of these grids. The objective behind our approach is enabling a more resilient and autonomous up-keeping of future grids. We investigate the application of evolving smart grids, which allow us to form dynamic clusters and allocate energy where needed. As an assumption, a smart energy gateway device is introduced which ensures the regulated use of authorized amounts of energy only. This allows active micro grid management by a coordinating entity. By incorporating an adapted leader election process into the bootstrap process of our model, we expect to enhance the grid's resilience in exceptional cases. Additionally, a proposed priority model enables the dissemination of scarce energy with preference until the grid has recovered.

ACKNOWLEDGEMENTS The work in this paper was performed in the context of the PolyEnergyNet project and partially funded by the Germany Federal Ministry for Economic Affairs and Energy (BMWi) under grant no. "0325737E". The author assumes responsibility for the content.

References

[AAH11] Arghavani, A.; Ahmadi, E.; Haghighat, A.T.: Improved bully election algorithm in distributed systems. In: ICIM 2011. pp. 1–6, Nov 2011.

[Fa10] Farhangi, H.: The path of the smart grid. Power and Energy Magazine, IEEE, 8(1):18–28, January 2010.

[Fa12a] Fang, Xi; Misra, Satyajayant; Xue, Guoliang; Yang, Dejun: Smart Grid – The New and Improved Power Grid: A Survey. IEEE COMMUN SURV TUT, 14(4):944–980, 2012.

[Fa12b] Fang, Xi; Misra, Satyajayant; Xue, Guoliang; Yang, Dejun: Smart Grid - The New and Improved Power Grid: A Survey. IEEE COMMUN SURV TUT, 14(4):944–980, 2012.

[Fa14] Farhangi, H.: A Road Map to Integration: Perspectives on Smart Grid Development. IEEE Power and Energy Magazine, 12(3):52–66, May 2014.

[GM82] Garcia-Molina, H.: Elections in a Distributed Computing System. Computers, IEEE Transactions on, C-31(1):48–59, Jan 1982.

[HHM11] Hashmi, M.; Hanninen, S.; Maki, K.: Survey of smart grid concepts, architectures, and technological demonstrations worldwide. In: ISGT Latin America. pp. 1–7, Oct 2011.

[Ho14] Horsmanheimo, S; Kamali, M; Kolehmainen, M; Neovius, M; Petre, L; Rönkkö, M; Sandvik, P: On proving recoverability of smart electrical grids. NFM, 8430 LNCS:77–91, 2014.

[LC02] Lee, Seok-Hyoung; Choi, Hoon: The Fast Bully Algorithm: For Electing a Coordinator Process in Distributed Systems. ICOIN, Springer-Verlag, London, UK, pp. 609–622, 2002.

[MV12] MVV Energie AG: , Model City Mannheim. http://www.modellstadt-mannheim.de/, 2012.

[Yu09] Yu, Cuibo; Gou, XueRong; Gou, Xuerong; Ji, Yang: Study on Supernode Election Algorithm in P2P Network Based upon District Partitioning. In: ICCSN. pp. 196–199, 2009.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 209

Order Preserving Encryption for Wide Column Stores

Tim Waage[1]

Abstract: Order-preserving encryption (OPE) allows encrypting without losing information about the order relation between the encrypted data items. Thus, the execution of compare, order and grouping operations can be done like on plaintext data. In particular it allows databases to do range queries over encrypted data, which is a useful feature especially for cloud databases that usually run in untrusted environments. Several OPE schemes have been proposed in the last years, but almost none of them are used in real world scenarios. While OPE was at least implemented for some SQL-based prototype systems before (e.g. [Po11, Tu13], our work identifies the practical requirements for utilizing OPE in existing NoSQL cloud database technologies. It also provides runtime analyses of two popular OPE schemes combined with two popular NoSQL wide column store databases.

Keywords: Order-preserving encryption, wide column stores, Apache Cassandra, Apache HBase

1 Introduction

Nowadays distributed data storage becomes more and more important due to the increased amount of data being produced every day, by private persons (e.g. in social media platforms) as well as by business or research. Especially modern web services have a high demand for availability, consistency, partition tolerance, performance, scalability and elasticity, that are at best difficult and expensive to achieve with traditional relational databases. NoSQL databases running in distributed cloud environments were made to meet those requirements. Unfortunately security was not a primary concern of their designers [Ok11]. Instead some sort of front end is assumed to take care of authentication, authorization, etc. Yet NoSQL databases, especially the sub-category of wide column stores (WCSs), are a key technology behind many popular platforms, e.g. HBase behind Facebook [Bo11], Cassandra behind eBay or BigTable behind almost every Google service [Ch08].

Encryption is always a handy tool when it comes to outsourcing data to such untrustworthy environments. Unfortunately it also limits the options for interacting with the data that was encrypted. Using only traditional encryption methods like AES or RSA is unfeasible, because such schemes do not preserve the plaintext properties, that database systems are relying on (see section 3). We focus on one of those properties: the order relation. Preserving it during encryption enables a database to perform tasks such as executing range queries like on plaintext data. Although order-preserving encryption (OPE) is an active field of research, the practical feasibility of most schemes is insufficient.

Our work makes the following contributions: (i) it evaluates the practical feasibility of two fundamentally different OPE schemes, namely [BCO11, KS14], in existing database sys-

[1] Georg-August-Universität Göttingen, Institut für Informatik, Goldschmidtstraße 7, 37077 Göttingen, tim.waage@informatik.uni-goettingen.de

tems, (ii) it identifies the special requirements of NoSQL WCSs regarding OPE encryption and (iii) it conducts a practical performance comparison of [BCO11, KS14] in combination with the currently most popular NoSQL WCS platforms [So] Apache Cassandra [LM10] and Apache HBase [Bo11] to assess their respective strength and weaknesses.

2 Feasibility and Security of OPE in Practice

OPE can be described briefly like follows. If D is the plaintext space, R the ciphertext space (both with an order defining relation $\leq$) and k is a secret key, then an OPE scheme is a function $f_k : D \rightarrow R$ for which: $x \leq y \Rightarrow f_k(x) \leq f_k(y)$ for all $x, y \in D$. [Ag04] were the first to define this problem of OPE and proposed a theoretical scheme to address it. However, achieving practical feasibility without sacrificing security is still a challenging task. We evaluate the feasibility of OPE schemes in database scenarios based on three criteria:

Ciphertext (im-)mutability. An OPE scheme is called *mutable*, if it requires its ciphertexts to be changed as more and more input gets encrypted. An example of this category is [KS14]. Its state is a set of ordered plaintext-ciphertext-pairs, initialized with $\{(-1,-1), (maxPlaintextValue, maxCiphertextValue)\}$. A new value is always inserted in the middle of the gap between the next lower and next greater already encrypted value. If these values are directly consecutive, there is no gap here and the state requires a re-balancing. In practice this results in the overhead of reading, re-encrypting and writing back the data to the database. However a practical advantage of mutable OPE schemes is their fast and simple decryption process, as can be observed in section 4. *Immutable* OPE schemes avoid the re-encryption overhead in the first place. Immutable means once a plaintext is encrypted, the corresponding ciphertext is final. An instance of this category is [BCO11]. It is based on the fact that any order-preserving function from $1...M$ to $1...N$ can be represented by a combination of M out of N ordered items. Thus, ciphertexts can be computed by sampling values according to the negative hypergeometric distribution. However, using immutable OPE schemes results in having less security than using a mutable schemes [PLZ13].

Need for additional data structures. OPE schemes producing mutable ciphertexts require additional data structures for storing their state (plaintext-ciphertext-mappings). That can be done using indices, trees, dictionaries etc., either on clientside (or at least a trusted enviroment), e.g. [KS14], or on serverside, e.g. [PLZ13, Ro15]. Note that in particular the maintenance of tree sturctures is very expensive to achieve for most (unmodified) database systems. Hence additional software components on serverside are often proposed for performance reasons, which makes practical implementations rather complex.

Need for additional architectural components. Client applications and database platforms normally do not have built-in mechanisms for (order-preserving) encryption. Thus additional components are required for both rewriting queries to make them work with the serverside data structures (as they might have to be altered for functioning with the OPE schemes) as well as for performing decryption and encryption itself. Usually those components have to reside in the trusted (clientside) environment (e.g. [Po11, Tu13]). However some OPE schemes even require components running co-located to the database server (e.g. [PLZ13]), which can not be considered practical due to the architectural overhead.

Regardless of these practical concerns the first formal security analysis of OPE [Bo09] proved that ideal security[2] with immutable ciphertexts can only be accomplished, if the ciphertext space size $|R|$ is exponential in the plaintext space size $|D|$, which is hard to achieve in practice. Security can still be improved, e.g. by modular plaintext shifting [BCO11] (easy to implement, but only a small security enhancement) or using fake queries to hide the query distribution [Ma15] (causing communication and computation overhead).

In practice ideal security can be achieved easier by OPE schemes producing mutable ciphertexts, because they do not have the requirement of a ciphertext space size being exponential in the plaintext space size. They also hide the frequency distribution of plaintext-ciphertext assignments much better, being able to achieve an almost uniform distribution (as shown e.g. by [Wo13]). Still, that also means dealing with unavoidable re-encryptions of (parts of) the ciphertext, that is already stored in the database. Recent schemes try to keep the number of such updates to a minimum [KS14] or take the burden of reassigning ciphertexts to components on serverside [PLZ13] to reduce at least communication costs.

An alternative approach to avoid re-encryption in the first place is pre-encrypting the whole plaintext space D in advance [Wo13, Li14]. The unfeasibility of such an approach can be illustrated using the following example: let D be defined by a common Integer datatype. Having a typical length of 32 bit, $|D|$ would be of size 2^{32}, which means 4.3 billion items would have to be pre-computed and stored (even if the majority is never used).

3 Wide Column Stores and OPE

Due to general working principles all WCSs share, OPE schemes have to satisfy certain requirements, depending on what kind of data they are supposed to encrypt. These principles can be roughly described as follows. While WCSs use tables, rows and columns like traditional relational (SQL-based) databases, the fundamental difference is that columns are created for each row instead of being predefined by the table structure. Every row has an identifier that is unique for the table (commonly referred to as "row key"). Data is maintained in lexicographic order by that key. As WCSs are distributed systems, ranges of such row keys serve as units of distribution. Hence similar row keys (and thus data items that are likely to be semantically related to each other) are always kept physically close together, in best case on neighboring sectors on disk, but at least on the same node of a cluster, so that reads of ranges require communication to a minimum number of machines. The smallest units of information are key-value-pairs with the key itself having multiple components. Thus, more formally WCSs can be considered sparse, distributed, multidimensional maps of the form $(table, rowkey, column, timestamp) \rightarrow value$ (see [Ch08]). Using OPE is essential for encrypting row keys to preserve the order of the rows and thus the way of data distribution. Only immutable OPE schemes should be used for that task, since mutable OPE schemes would cause row keys to change over time. Because row keys are used for coordinating distribution, that would result in changing the data's physical position inside the database (cluster), which is prohibitively expensive. That is why most WCSs do not

[2] meaning "IND-OCPA": ciphertexts reveal nothing, but their order

even support changing row keys at all. However, mutable OPE schemes can be used for the column data itself to gain more security and performance. Since disk access and memory management in WCSs are performed at column family level, it is advisable to build the indices in the same way. In particular having one global index for all the system's order-preserving encrypted data items should be avoided, because it poses a performance bottleneck and a security threat (a compromised index would affect the entire database).

4 Experiments

For our experiments we inserted up to 10.000 uniformly distributed and randomly created numeric values into Cassandra and HBase using two OPE schemes that are practical, based on the criteria we introduced in section 2. Firstly, we use [BCO11] for which there are no alternatives, when immutable ciphertexts and having no state is desired. Secondly, we use [KS14], because its need for additional data structures is minimal (only a clientside index). Both schemes do not require further architectural components. We use a plaintext space size $|D|$ of 25 bit and a ciphertext space size $|R|$ of 32 bit, which satisfies the recommendations of the authors of both schemes. While for [BCO11] the order of insertion does not matter, there are three cases to consider for [KS14]. The best case is when all elements of a perfectly balanced binary search tree are inserted in pre-order traversal order. The average case is a uniform input distribution. The worst case is inserting pre-sorted values. We use local installations to avoid network effects, as we want to measure the computation time of the schemes in combination with the insertion speed of the databases. All implementations were done in Java 8. We ran our experiments on an Intel Core i7-4600U CPU @ 2.10GHz, 8GB RAM, a Samsung PM851 256GB SSD using Ubuntu 15.04.

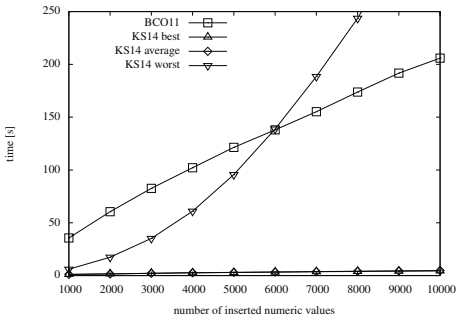


Fig. 1: Time needed for encryption with increasing data set size in Apache Cassandra

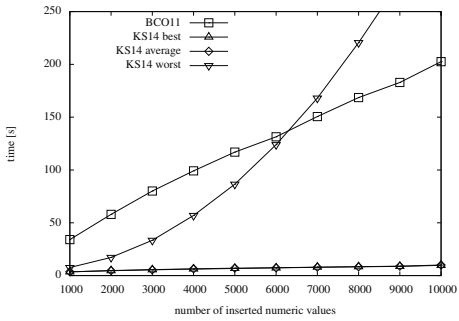


Fig. 2: Time needed for encryption with increasing data set size in Apache HBase

Figure 1 and 2 present the results, showing the average of five measurements. Both databases perform equally well for [BCO11], but [KS14] in its best and average case is much less computationally expensive and thus faster by a factor of roughly 45 using Cassandra and still 20 using HBase, which means Cassandra is twice as fast in those scenarios. This can be explained by two reasons. Firstly, [KS14] is so fast that the database systems pure insertion time requires a significant share in the the overall process of encrypting and inserting (which is not the case for the expensive encryption process of [BCO11]). Secondly,

[KS14] needs no re-encrypting for 10.000 values with the given sizes for plaintext and ciphertext space (in fact the first re-encryption occurs after inserting ca. 12.000 values on average). With Cassandra being optimized for writes it takes advantage of both. It can be further observed that [KS14] gets prohibitively slow when encrypting ordered values, starting to perform even worse than [BCO11] after 6000 insertions (which already require over 300 re-encryptions). Hence not only writing but also reading performance matters. Interestingly in this case HBase is always 12-15% faster than Cassandra, which seems to reflect the fact, that while Cassandra is optimized for writes, HBase is optimized for reads. However Cassandra performs equally or better in all other disciplines.

Because decrypting is very simple, we do not elaborate on it in the same level of detail as we did for encrypting. It does not even involve the database platforms. In [KS14] it is just a lookup in the clientside index which takes less than 1 ms. In [BCO11] it requires rather expensive computations, taking 68 ms on average due to Javas BigInteger class. This performance can be improved by using standard Integer types or caching already decrypted values, which results in the same speed ($<$ 1 ms) as for using [KS14].

5 Related Work

So far there is not much work using OPE with real world technologies. The most popular example surely is “CryptDB” [Po11] utilizing the immutable scheme of [Bo09], tweaked by operating with a binary search tree and caching in the background. Another system for executing queries over encrypted data is “Monomi”, also using [Bo09] for OPE. Both approaches are designed for working with SQL-based systems.

6 Conclusion and Future Work

We discussed how OPE can be used in NoSQL WCSs and quantified the performance of two OPE schemes on the two currently most popular platforms. As we already did the same for a couple of schemes for searchable encryption [WJW15], our next goal is to build a seamless integrating proxy client similar to “CryptDB” for executing more sophisticated queries on encrypted WCS databases.

Acknowledgement: This work was funded by the DFG under grant number WI 4086/2-1.

References

[Ag04] Agrawal, Rakesh; Kiernan, Jerry; Srikant, Ramakrishnan; Xu, Yirong: Order preserving encryption for numeric data. In: Proceedings of the 2004 ACM SIGMOD International Conference on Management of Data. ACM, pp. 563–574, 2004.

[BCO11] Boldyreva, Alexandra; Chenette, Nathan; O’Neill, Adam: Order-preserving encryption revisited: Improved security analysis and alternative solutions. In: Advances in Cryptology–CRYPTO 2011, pp. 578–595. Springer, 2011.

[Bo09] Boldyreva, Alexandra; Chenette, Nathan; Lee, Younho; O'Neill, Adam: Order-preserving symmetric encryption. In: Advances in Cryptology-EUROCRYPT 2009, pp. 224–241. Springer, 2009.

[Bo11] Borthakur, Dhruba; Gray, Jonathan; Sarma, Joydeep Sen; Muthukkaruppan, Kannan; Spiegelberg, Nicolas; Kuang, Hairong; Ranganathan, Karthik; Molkov, Dmytro; Menon, Aravind: Apache Hadoop goes realtime at Facebook. In: Proceedings of the SIGMOD International Conference on Management of Data. ACM, pp. 1071–1080, 2011.

[Ch08] Chang, Fay; Dean, Jeffrey; Ghemawat, Sanjay; Hsieh, Wilson C; Wallach, Deborah A; Burrows, Mike; Chandra, Tushar; Fikes, Andrew: Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS), 26(2):4, 2008.

[KS14] Kerschbaum, Florian; Schröpfer, Axel: Optimal average-complexity ideal-security order-preserving encryption. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, pp. 275–286, 2014.

[Li14] Liu, Zheli; Chen, Xiaofeng; Yang, Jun; Jia, Chunfu; You, Ilsun: New order preserving encryption model for outsourced databases in cloud environments. Journal of Network and Computer Applications, 2014.

[LM10] Lakshman, Avinash; Malik, Prashant: Cassandra: a decentralized structured storage system. ACM SIGOPS Operating Systems Review, 44(2):35–40, 2010.

[Ma15] Mavroforakis, Charalampos; Chenette, Nathan; O'Neill, Adam; Kollios, George; Canetti, Ran: Modular Order-Preserving Encryption, Revisited. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. ACM, pp. 763–777, 2015.

[Ok11] Okman, Lior; Gal-Oz, Nurit; Gonen, Yaron; Gudes, Ehud; Abramov, Jenny: Security issues in nosql databases. In: Trust, Security and Privacy in Computing and Communications, 2011 IEEE 10th International Conference on. IEEE, pp. 541–547, 2011.

[PLZ13] Popa, Raluca A; Li, Frank H; Zeldovich, Nickolai: An ideal-security protocol for order-preserving encoding. In: IEEE Symposium on Security and Privacy. pp. 463–477, 2013.

[Po11] Popa, Raluca Ada; Redfield, Catherine; Zeldovich, Nickolai; Balakrishnan, Hari: CryptDB: protecting confidentiality with encrypted query processing. In: Proceedings of the 23rd ACM Symposium on Operating Systems Principles. ACM, pp. 85–100, 2011.

[Ro15] Roche, Daniel; Apon, Daniel; Choi, Seung Geol; Yerukhimov, Arkady: POPE: Partial order-preserving encoding. Technical report, Cryptology ePrint Arch. 2015/1106, 2015.

[So] SolidIT: DB-Engines Ranking. `http://db-engines.com/en/ranking`, 19.01.2016.

[Tu13] Tu, Stephen; Kaashoek, M Frans; Madden, Samuel; Zeldovich, Nickolai: Processing analytical queries over encrypted data. In: Proceedings of the VLDB Endowment. volume 6. VLDB Endowment, pp. 289–300, 2013.

[WJW15] Waage, Tim; Jhajj, Ramaninder Singh; Wiese, Lena: Searchable Encryption in Apache Cassandra. In: Proceedings of the 8th Symposium on Foundations and Practice of Security (FPS). Springer, 2015.

[Wo13] Wozniak, Sander; Rossberg, Michael; Grau, Sascha; Alshawish, Ali; Schaefer, Guenter: Beyond the ideal object: towards disclosure-resilient order-preserving encryption schemes. In: Proceedings of the 2013 ACM workshop on Cloud computing security. ACM, pp. 89–100, 2013.

International Workshop on
Security, Privacy and Reliability of Smart Buildings

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 217

Software Security Requirements in Building Automation

Friedrich Praus[1], Wolfgang Kastner[2], Peter Palensky[3]

Abstract: With today's ongoing integration of heterogeneous building automation systems, increased comfort, energy efficiency, improved building management, sustainability as well as advanced applications such as active & assisted living scenarios become possible.

Obviously, the demands – especially regarding security – increase: Secure communication becomes equally important as secure software being executed on the devices. While the former has been addressed by standardization committees and manufacturers, until recently no scientific research is available, that targets the problem of secure control applications in this domain. No attack model has been defined, no security measures have been recommended, existing measures from other domains are either too expensive or time intensive to deploy, cannot be trivially applied to or do not cover specific demands and constraints of the building automation domain.

This paper provides an extensive survey of the security requirements for distributed control applications and analyzes software protection methods. An architecture tackling the problem on how to secure software running on different device classes and preventing attacks on smart homes and buildings is briefly introduced at the end.

Keywords: Secure Software, Security Process, Secure Control Applications, Smart Homes, Security, Building Automation

1 Introduction and Motivation

In order to provide secure Building Automation Systems (BASs), comprehensive measures need to cover communication as well as device security. Mechanisms tailored to the use in Building Automation Networks (BANs) that counteract communication and network attacks are presented in [Gr10]. An overall device security needs to deal with software, side-channel, and physical attacks. An extensive survey on the latter two and a short discussion of countermeasures can be found in [KS04]. Until recently, no scientific research is available that extensively targets security requirements and software attacks in the BAS domain.

Figure 1 briefly describes the life cycle of a BAS and involved stakeholders over the building lifetime. A building owner defines the requirements and instructs a planner to plan the building. The system integrator selects and commissions the devices and an installer deploys them in the building. The facility manager finally maintains the building. Basically, security is essential in all steps, for all stakeholders and during the complete lifetime. To

[1] University of Applied Sciences Technikum Wien, Höchstädtplatz 6, 1200 Wien, praus@technikum-wien.at,
This work was partly funded by the City of Vienna, under grant number MA23-Projekt 15-03.
[2] TU Wien, Automation Systems Group, Treitlstr. 1-3/4, 1040 Wien, k@auto.tuwien.ac.at
[3] TU Delft, Intelligent Electrical Power Grids, Mekelweg 4, 2628 CD Delft, P.Palensky@tudelft.nl

focus on secure Control Applications (CAs), the following use cases are identified as being security critical (blue and bold-italic markings in Figure 1): Starting from secure CA development, committees need support in standardizing mechanisms, frameworks and generic policies, while CA manufacturers need support in creating CAs. System integrators and facility managers need support in adapting security policies during CA operation.

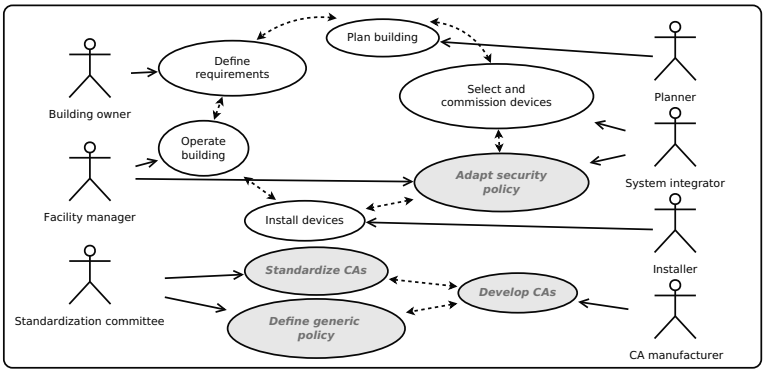


Fig. 1: Life Cycle of a Building Automation System and Involved Stakeholders

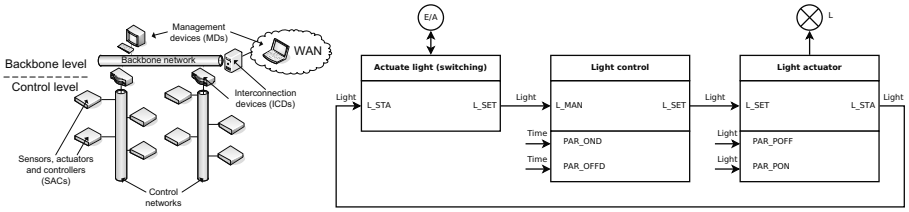
Today's software for smart home and building devices lacks adequate security mechanisms. Irrespective of the used technology, no sound protection against software attacks is deployed, thus enabling adversaries to successfully attack those devices. Existing protection techniques from other domains (e.g. the Information Technology (IT) domain) are insufficient and not applicable to BASs due to different functional and non-functional requirements. Thus, a secure architecture being adaptable to all common BAS standards needs to be established. This architecture needs to cover BAS specific constraints, provide a security policy and a secure software environment. Besides, mechanisms for attack detection are needed [Pr15].

The paper is structured as follows. Section 2 analyzes domain constraints in distributed CAs and the structure of modern BASs and their devices. Based on a short analysis of CA security in current building automation standards and technologies and a software security threat and risk analysis, security requirements are identified (cf. Section 3). Existing software protection techniques are briefly investigated and evaluated for BASs with respect to their applicability and implied security gain (cf. Section 4). Then, a first concept for secure and distributed CAs, describing how to fulfill the demands for security-critical smart homes and buildings is presented (cf. Section 5). Finally, an outlook on the evaluation and future work is described in Section 6.

2 Control Applications in Building Automation Systems

Today's BAS are implemented following a two-tier architecture (cf. Fig. 2a). It consists of a control network and a common backbone which together form the BAN. Sensor Actuator and Controllers (SACs) are located at the control level. Representatives of this device class interact directly with the physical environment and are responsible for data acquisition and

for controlling the behavior of the environment. InterConnection Devices (ICDs) provide an interconnection between network segments or remote access to foreign networks. Management Devices (MDs) are used to configure and maintain a BAS.



(a) Building Automation Network [GPK10]

(b) Use Case ⊗: Distributed Control Application: VDI 3813-3: D-1-2 Lighting Control Manual with Time-Controlled Switching Off (Stairwell Light)

Fig. 2: Control Applications in Building Automation Systems

While the functionality of system components (i.e. ICDs and MDs) is usually fixed, SACs are highly customizable and manifold. Thus, in the BAS domain typically the approach exists to customize generic "template" network nodes with application specific hardware. Universally designed base platforms consisting of MicroController Units (MCUs) and network interfaces are used in conjunction with application specific components (e.g. switches, temperature sensors) to form a particular system. Similarly, the software is split into a generic Operating System (OS) or system software providing basic functionality and a customizable CA dealing with the specific hardware. While the former is usually fixed and non replaceable, the latter is implemented by the device manufacturer and may be downloaded by the system integrator, even after installation of the device. Thus, a CA is a configurable software being executed on a SAC with the purpose to control a process at the control level. Distributed CAs communicate via the BAN and implement a particular function of a BAS.

A common way to model distributed CAs and their application models is with the help of Function Blocks (FBs). Sensor functions convert physical quantities to output information which in turn serves as input to application or actuator functions. Actuator functions convert input information obtained through the BAN to physical quantities. Application functions represent the functionality to be achieved by means of automation and control. To be able to describe the security concept presented in this paper in a better way, the following use case ⊗ is defined according to VDI 3813-3 and will be used throughout the remaining sections. Figure 2b shows a minimalistic light actuator with delayed on/off behavior being used for a stairwell light.

Use case ⊗ *Stairwell light: When pressing the "on" button* `E` *of the light sensor, the attached light control immediately triggers the light actuator to switch on lamp* `L`*. Upon pressing the "off" button* `E` *of the light sensor, the attached light control waits the time configured with the off delay parameter* `PAR_OFFD` *and then triggers the light actuator to switch off the lamp* `L`*. The output* `L_STA` *of the light actuator is used as feedback to the actuate light sensor, to be able to display the current status using output* `A`*.*

Today's open BAS technologies (i.e. BACnet, EnOcean, KNX, LonWorks and ZigBee) achieve interoperability each by standardizing their own CA model. As shown in [Pr15], these application models differ significantly even for simple use cases such as a stairwell light or an individual room control. A transparent (i.e. translation/gateway-free) communication across technology borders is likewise impossible, as common security mechanisms are missing. The actual functionality of a BAS, however, is always similar and most of the traditional functions can likewise be realized by any of these technologies. An overall software security in smart homes and buildings can only be established, if the required measures are applicable to all common standards and technologies. Therefore, to be able to develop a secure CA architecture being adaptable to all different standards and technologies, a consistent application model is required. Security can then be investigated for this model and appropriate measures can be derived and developed for the different technologies and standards.

3 Control Application Security

The ultimate goal of an adversary is to gain unauthorized access to control level functions by manipulating the software being executed on BAS devices. A possible attack scenario is an adversary trying to hack a door switch to illegally access a building or to neutralize a light actuator by crashing its CA. Such attacks can either be performed remotely via the network or locally, exploiting threats in a device's interface. First, an adversary may directly access SACs to manipulate the behavior of the hosted CAs by changing configuration parameters (e.g. setpoint), the control logic (e.g. algorithm), or the control data (e.g. output value). Second, an adversary may attack the application running on the ICD to get access to the data passing through the ICD. As ICDs may also provide an interconnection to foreign public networks (e.g. the Internet), an ICD can also be misused as access point to launch further attacks via the BAN. Finally, an adversary may attack a MD by manipulating the operator software and also impersonate a MD. The privileges of the compromised device can then be misused to gain management access to SACs or ICDs.

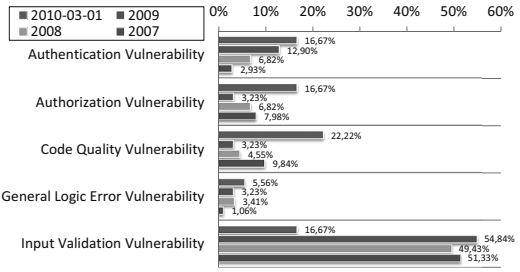
3.1 An Overview of Security in BAS

To identify the security threat in current installations, recent research [Pr15] targeted the question, whether and how many BASs based on BACnet and KNX are openly connected to the Internet and what security measures are currently implemented. A worldwide IPv4 address range scan for BACnet/IP and KNXnet/IP services has been carried out in 2014 by the author. A total of 17.259 vulnerable BAS installations have been detected (cf. Table 3a). BACnet is being widely used in the US and Canada whereas KNX is very popular in Europe. The installations ranged from business parks and towers, high schools, shopping plazas, water pollution control stations, fire stations, churches to smart homes with control of private saunas.

To be able to provide secure CAs in BASs, it is first necessary to identify the threats to CA software and analyze possible vulnerabilities. The Open Web Application Security

Country	BACnet	Country	KNX
US	8989	Germany	627
Canada	2296	Netherlands	522
Finland	282	Spain	332
Australia	271	France	244
Spain	231	Austria	220
Total	13964	Total	3295

(a) Vulnerable BACnet and KNX Installations



(b) Vulnerabilities Breakdown 2010-03 – 2007

Fig. 3: Control Application Security (Top 5 Countries and Categories)

Project Top 10[3] provides such an analysis covering over 500,000 vulnerabilities tailored to web application security. Although being slightly different from the security of CAs, a categorization can nevertheless be derived, if it is assumed similar security flaws are present. Based on this categorization, an analysis of the commonness of vulnerability types can be performed. Figure 3b shows a breakdown of the top 5 vulnerabilities being openly available at the US-CERT Vulnerability Notes Database[4]. All entries of the years 2007 to March 2010 (632 in total) have been analyzed, categorized and counted [Pr15]. Although being rather outdated, the numbers still give an adequate overview of the commonness of vulnerability types.

As shown in this section, security awareness in the BAS domain is missing and today's CAs need to be considered as insecure. Thousands of installations are being directly connected to the Internet, allowing an adversary to attack them.

3.2 Security Requirements

Due to the extreme broadness of threats and vulnerabilities to CAs, the software attack model is defined as follows: Any (malicious) CA, irrelevant whether it originates from trusted or non-trusted sources, being run on BAS devices may exploit weaknesses in security schemes and system implementations, intentionally or unintentionally. Accidental programming flaws in CAs may be present just like software being intentionally infected by trojans. Adversaries may use these manifold possibilities to access control level functions they usually are not allowed to.

Based on this attack model, security requirements dedicated to CAs are formulated. They are derived out of security research in e.g. industrial communication systems [Dz05], embedded systems [Ra04] or cyber-physical systems.

[3] `http://www.owasp.org/`, Last access: 2016/02/02
[4] `http://www.kb.cert.org/vuls`, Last access: 2010/04/03

Functional Requirements (FRs) are directly related to the security considerations for CAs. The utmost requirement is to prevent software attacks on CAs and, if not possible, at least detect those attacks. The following FRs can be derived to achieve this goal:

***FR–memory access*:** Considering the execution of a CA on a SAC, the memory access must be controlled. On the one hand, a CA must not be allowed to access arbitrary memory locations to e.g. prohibit, that a malicious CA subverts any security mechanism. On the other hand, a secure storage of protected data must be possible. To put it differently, information such as configuration parameters or cryptographic keys invisible and unaccessible to the CA need to be stored on the SAC to provide the basis for a secure system. Vulnerabilities (e.g. memory corruption via buffer and format string overflows or code injection) caused by side effects have to be prevented.
***FR–low level functionality access*:** The same way it must be possible to limit the actions and allowed operations (e.g. access to low-level function calls) a CA can perform with respect to

- Access rights: Is a CA allowed to call a particular function or not? Note, that often a generic system software is deployed on SACs with far more capabilities and functions than a simple CA may need. Hence, it is desirable to limit the allowed operations for a SAC.
- Parameters: Likewise the parameters of a function call need to be limited so that e.g. the present value of a Datapoint (DP) does not exceed a critical value.
- Execution time: The point in time when a function is called is an additional constraint to monitor. Not only the actual instance, but also the invoking frequency is critical for some applications.
- Domain constraints: Dependencies between function calls, which can be seen as domain constraints, are a further critical issue. Consider e.g. an Heating, cooling, Ventilation, and Air Conditioning (HVAC) application, where it is not desirable to simultaneously switch on the heating and the cooling function.

***FR–protection of environment*:** CAs must neither destruct the hardware or waste resources intentionally nor due to programming flaws (e.g. wear out of a flash memory or exhaust battery power).
***FR–communication relationship*:** CAs have a defined (static) communication relationship. Being readily configured, it is known which CAs need to communicate and which CAs do not need to communicate. A simple light switch, for instance, must not be hacked and abused to open a security door. This communication relationship needs to be considered in security mechanisms.
***FR–availability*:** Availability needs to be guaranteed. Denial of Service (DoS) attacks need to be prevented or detected.

Organizational Requirements (ORs) cover the special environmental conditions required for developing secure CAs in BAS.

***OR–limited resources*:** Due to cost efficiency and form factor, SACs are normally embedded devices with limited system resources (e.g. memory, processing power) that rely on either bus or battery power. Security mechanisms are computationally intensive and must not exceed the available device's processing resources (processing gap) and power

resources (battery gap) [Ra04]. The overhead imposed by these mechanisms needs to be reasonably small. Therefore, a suitable balance between a required level of security and available resources has to be found (”good enough security”).

***OR–development*:** CA development has to be simple and secure by design so that even security unaware developers are able to design secure CAs. This is especially important for the BAS domain, since engineers are experts in the field of automation but not in the field of security. Therefore, a two level concept with a dedicated system software and a CA, as already present in KNX or LonWorks, needs to be supported.

***OR–high level language support*:** High-level programming languages (e.g. Java) need to be supported such that the desired control logic and behavior can be obtained more easily. CA development is also simplified, since the application programmer does not have to cope with details such as a hardware specific system software or the communication protocol.

***OR–long lifetime*:** BASs have to be kept operable for years or even decades. Due to this long lifetime, such systems obviously have to undergo maintenance during runtime in order to keep them operable. With the complexity of the CA software increasing, it also must be assumed that not all implementation flaws can be detected in the development phase. Since these may result in security vulnerabilities, a secure update mechanism is beneficial. Such a mechanism should allow the distribution of system software patches and secure download and replacement of CAs in an easy and secure manner.

***OR–scalability*:** Since BASs can consist of hundreds or even thousands of devices, appropriate scalability of security mechanisms is essential. For instance, key distribution schemes which routinely require physical access to the individual devices are not feasible in large networks. Therefore, services must be provided which assist in performing these tasks.

***OR–network technology*:** Security mechanisms need to be geared towards the different requirements in BANs regarding the used network technology. While in the IT world Internet Protocol (IP) based network protocols are dominant, the use of IP networks in BANs is reserved to the backbone level. At the field level, predominantly non-IP fieldbus are used. Besides, control data typically transmitted in BANs have a small volume (in the order of bytes) with perhaps soft real time requirements (e.g. reaction time in a lighting system).

***OR–compatibility*:** The integration of a security extension into an established BAS is preferable to create an entirely new system. Such an approach allows to leverage the existing base of available components for parts of the system where security is not (yet) a requirement. This allows a smooth transition until devices supporting the security extension become widely available. It also offers an economical upgrade path for existing installations.

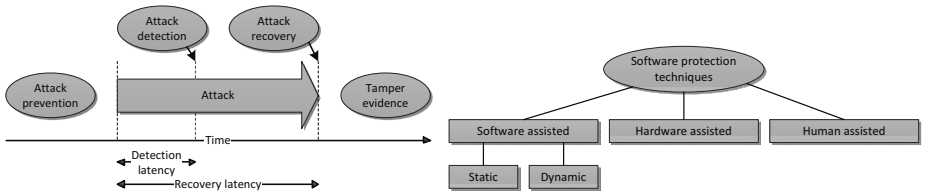
***OR–physical access*:** In BANs, devices often operate in untrusted environments where physical access (e.g. an intrusion alarm in a public building or a wireless sensor network [GH09]) is given. Therefore, it has to be assumed that a short time physical access to devices and networks cannot be avoided. Such attacks have to be detected by a security system.

***OR–usability*:** Usability of security measures has to be provided, when these systems are installed. On the one hand, this implies that it has to be possible to deploy them as easily as possible. In the best case, users do not even notice, that a security measure is enabled.

At least, education and guidelines (e.g. secure password guidelines) need to be provided for support. On the other hand, this requirement also covers protection against social engineering attacks.

As shown in this section manifold requirements need to be considered to be able to provide secure CAs.

4 Software Protection Techniques



(a) Attack Model (b) Software Protection Techniques

Fig. 4: Software Protection Techniques [Pr15]

The ideal software protection technique allows to fully prevent vulnerabilities and hinders attacks to SACs, ICDs and MDs, no matter whether the attack pattern is already known or not. To minimize performance overhead, it is applied only during compile time, or at least does not have any performance overhead during runtime. Besides, it does not require updates and scales well. Considering today's available methods, it is however hardly possible to fulfill all these requirements at the same time due to e.g. limited system resources.

This section focuses on methods trying to prevent or at least detect attacks within a reasonable detection latency (cf. Figure 4a). Possible software protection techniques (cf. Figure 4b) are briefly discussed with respect to the security requirements formulated in the previous section and applicability to the device classes in BAS.

Software assisted protection techniques can be split into static and dynamic methods.

Static methods are applied during compile or development time, respectively. Hence, they can prevent attacks at a point in time, where appropriate countermeasures or bug fixes can be applied without interfering with running software. However, they cannot detect all possible vulnerabilities without actually executing a piece of software. Besides, Static Code Analysis (SCA), Code-Signing (CS) and Proof-Carrying Code (PCC) have to be performed on every code change. Nevertheless, they are assumed to be easily applicable with respect to resources of the target system because they are only used at compile time. CS can be quite effective to prevent the installation of arbitrary CAs by simply refusing to execute unsigned or not properly signed code. Watermarking (WM) applied to software protects against illegitimate modifications and tampering by its users. The universal applicability of PCC to BAS is questionable since the generation and encoding of proofs for complex security policies are nontrivial tasks and have to be performed on every code change.

−: not applicable, ∼: applicable with restrictions, *p*: prevent—*d*: detect—+: applicable

	method or BAS	*FR–memory access*	*FR–low level functionality access*	*FR–protection of environment*	*FR–communication relationship*	*FR–availability*	*OR–limited resources*	*OR–development*	*OR–high level language support*	*OR–long lifetime*	*OR–scalability*	*OR–network technology*	*OR–compatibility*	*OR–physical access*	*OR–usability*	SAC	ICD	MD
static	SCA	−	−	∼	−	−	+	∼	+	−	−	+	∼	−	−	∼	+	−
software	CS	−	−	−	−	−	+	∼	+	+	+	+	∼	−	−	+	+	+
methods	WM	−	−	−	−	−	+	−	+	−	+	+	∼	−	−	+	+	+
	PCC	−	−	∼	−	−	+	−	+	+	−	+	∼	−	−	−	∼	∼
	SIDS	−	−	*d*	*d*	−	∼	∼	+	−	−	+	+	−	−	+	+	∼
	AIDS	−	−	*d*	*d*	*d*	−	+	+	+	+	+	+	+	∼	+	+	∼
dynamic	SMT	−	−	∼	−	−	∼	−	+	−	−	+	∼	−	−	∼	+	−
software	SB	*p*	*p*	*p*	*p*	−	∼	+	+	+	+	+	∼	−	+	+	∼	+
methods	SCC	−	−	−	−	−	∼	−	+	−	−	∼	∼	−	−	−	+	−
	ASC	−	−	−	−	−	∼	−	+	−	−	+	∼	−	∼	∼	+	+
	OS	*p*	*p*	*p*	*p*	−	−	+	+	+	+	+	∼	−	+	−	−	+
	CP	−	∼	−	∼	−	+	−	+	−	−	+	∼	+	−	−	−	+
hardware	PP	*p*	*p*	−	−	−	+	+	+	−	−	+	∼	+	−	−	−	+
supported	HA	−	−	−	−	−	+	+	+	−	+	+	∼	−	+	+	+	+
	CPUEX	−	−	−	−	−	+	+	−	−	+	+	∼	−	+	+	+	+
human	IAC	∼	−	*p*	−	−	+	−	+	−	−	+	∼	−	−	−	∼	−
	FV	−	−	*p*	−	−	+	−	+	−	−	+	∼	−	−	−	∼	−

Tab. 1: Comparison of Software Protection Techniques with Respect to Security Requirements and Applicability to Sensors, Actuators and Controller Devices, Interconnection Devices, and Management Devices

Dynamic methods implicate a larger performance overhead than static ones, since additional processing has to be performed during runtime. In addition, special care has to be taken, that they are not bypassed by an adversary. Signature based Intrusion Detection Systems (SIDSs) offer a high attack detection accuracy. They, however, cannot detect new attacks and are vulnerable to attack variations such as worms, that alter their own code base on succeeding executions. SIDSs are not well suited as they depend on a usually large database and require constant updates, which would be difficult in case of SACs and ICDs. Anomaly based Intrusion Detection Systems (AIDSs) in contrast also allow to detect novel intrusions, which are not yet present in the database of an Intrusion Detection System (IDS). They are, however, not able to distinguish between natural changes of the monitored system and attacks. AIDS, Software Monitoring Techniques (SMT) as well as Self Checking Code (SCC) may be efficiently implemented and could therefore be quite appropriate. SMT at least requires hardware support for context switches. Attack Specific Countermeasures (ASCs) can also work well in many cases, but might not be applicable due to differing processor and memory architectures on SACs and ICDs. The applicabil-

ity of Sandboxes (SBes) strongly depends on the overhead imposed by their feature sets. While a reduced and lightweight SB could easily be deployed to SACs, an architecture like the full Java Virtual Machine (JVM) with its vast execution and security mechanisms imposes a big overhead. While in the IT world and thus also for MDs OSs typically limit what an application is allowed to do, the targeted MCUs being deployed to SACs and ICDs do not provide the necessary hardware support (e.g. lack of memory management units to separate the address spaces of different processes).

Hardware assisted methods use dedicated hardware for security checks (e.g. Co-Processor (CP), Physical Partitioning (PP), Harvard Architecture (HA), CPU EXtension (CPUEX)) and allow to lower the imposed performance overhead in contrast to pure software based methods. However, hardware supported methods requiring additional components cannot be cost effectively deployed to SACs and recent research also demonstrates that these methods may also be bypassed [Ro12]

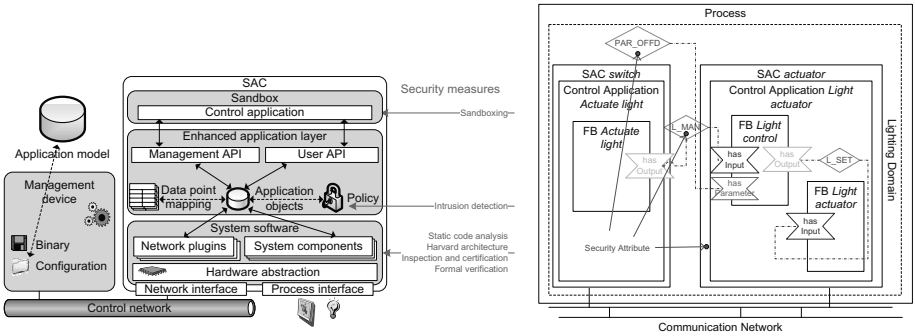
Human assisted methods rely on human expertise during CA development. Inspection And Certification (IAC) performed by humans may eliminate a lot of possible attacks, but requires extensive knowledge by the auditing person, is time consuming, expensive and error-prone. Thus, it does not scale at all and may limit flexibility, since it is only feasible to be applied to code, which does not change frequently. Formal Verification (FV) is the most tenable method for providing security constraints. If security attributes can be represented in a formal way, provers can guarantee that these attributes are met. However, applying FV to real life software or even an OS is a very hard task and only two approaches are known for now that allow Common Criteria EAL6+ certification [SAIC08, Kl09].

Hybrid methods try to combine the advantages of different software protection techniques. Thus, they can provide more powerful protection and overcome limitations of the software, hardware and human assisted methods mentioned before. Until now however, no reliable and secure approach for BASs is available. It is not clear, which combinations of software protection techniques seem reasonable and fulfill the security requirements.

5 Secure Control Application Architecture

A secure architecture being adaptable to all common BAS standards needs to cover BAS specific constraints and be capable of detecting possible attacks. Thus, only hybrid software protection mechanisms can provide an overall CA security.

The resulting secure CA architecture (cf. Figure 5a) consists of four parts: First, a generic application model is defined, being required to develop a secure system being used for the different BASs. It separates generic information of BASs from an installation dependent one. This is achieved by the definition of the abstract model (e.g. the abstract BAS device description) and concrete instances therefrom (e.g. a BAS device instance representing a particular technology with specific parameters). Additionally, protocol-specific and domain-specific knowledge (e.g. BAS specific vocabulary, security attributes) are part of the model. All configuration and management tasks and definition of a security policy can now be performed directly on the abstracted representation and be automatically distributed to the different underlying technologies.



(a) Architecture of Sensor Actuator and Controllers (b) Application Model and Security Policy for Use Case ⊗

Fig. 5: Secure Control Application Architecture

Second, a security policy based on security attributes allows a formal way to formulate security requirements. This global policy states, whether the condition of a BAS is security critical and violates some defined constraints or not. For executing this policy it can be split down to involved present values of DPs, where security requirements for the conditions derived from the policy can be defined, formulated and finally evaluated. An instance of the application model and an example security policy with security attributes for use case ⊗ is shown in Figure 5b.

Third, a software environment to securely execute CAs and enforce the security policy is needed. As outlined in Figure 5a, it consists of three major components, each imposing an additional security barrier to the overall security and limiting possible security threats: A system software provides controlled access to system resources. It encapsulates hardware specific details, the network protocol stack, the process interface as well as any further system components and offers clean interfaces for the enhanced application layer. An enhanced application layer stores the application objects, their DP mappings as well as the security policy for the CA. A Sandbox executes the CA in a controlled way and is also designed to support its rapid development. It interfaces the system software via the enhanced application layer and provides a clear abstraction of the underlying hardware and software by providing an object-oriented access (using e.g. the Java programming language). The application designer can thus focus on the application development. Finally, methods to detect possible attacks (e.g. DoS) and violations of the security policy are needed.

6 Summary and Future Work

To summarize, the following hybrid software protection mechanisms are deployed to provide security of the presented architecture SCA, IAC and code reviews are performed on the system software, which provides an abstraction and layering to ease CA development. A combination of an AIDS and a SIDS based upon a security policy and a generic application model are used to detect and prevent attacks. Uncircumventable sandboxing of

CAs is performed to protect the system outside of the SB from attacks. WM could also be deployed to only execute signed CAs, if desired.

Details on the proposed architecture and a validation of its feasibility have recently been shown in [Pr15]. They will be published in future work: First, the process of how to implement secure CAs will be described and it will be shown, how the generic architecture can be instantiated, i.e. how the CA can be modeled and how the policies can be adapted. On the basis of use case $\otimes$, prototypes will be implemented for the different device classes to evaluate and test the stability with respect to memory consumption, performance, and security. To further evaluate the presented concept, a discussion on how it can be used to enable security in today's already existing BASs will follow. Attack detection and prevention become possible, if appropriate devices are installed. Concluding, an exhaustive discussion will evaluate, that all functional requirements are fulfilled by means of the concept. Some organizational requirements still need to be considered, when implementing real live installations. Thus, it will be proven, that the developed secure CA architecture can be implemented on the devices typically found in BAS. Besides, it fulfills the requirements for secure CAs and is able to prevent or at least detect attacks.

References

[Dz05] Dzung, D.; Naedele, M.; Von Hoff, T.P.; Crevatin, M.: Security for Industrial Communication Systems. 93(6):1152–1177, 2005.

[GH09] Gungor, V.; Hancke, G.: Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches. 56(10):4258–4265, 2009.

[GPK10] Granzer, Wolfgang; Praus, Friedrich; Kastner, Wolfgang: Security in Building Automation Systems. IEEE Transactions on Industrial Electronics, 57(11):3622–3630, Nov.'10.

[Gr10] Granzer, Wolfgang: Secure Communication in Home and Building Automation Systems. PhD thesis, Vienna University of Technology, February 2010.

[Kl09] Klein et al.: seL4: Formal Verification of an OS Kernel. In: SOSP '09: Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles. ACM, New York, NY, USA, pp. 207–220, 2009.

[KS04] Koeune, François; Standaert, François-Xavier: A Tutorial on Physical Security and Side-Channel Attacks. In: Foundations of Security Analysis and Design III, FOSAD 2004/2005 Tutorial Lectures. pp. 78–108, 2004.

[Pr15] Praus, Friedrich: Secure Control Applications in Smart Homes and Buildings. PhD thesis, Technische Universität Wien, November 2015.

[Ra04] Ravi, Srivaths; Raghunathan, Anand; Kocher, Paul; Hattangady, Sunil: Security in Embedded Systems: Design Challenges. Transactions on Embedded Computing Systems, 3(3):461–491, 2004.

[Ro12] Roemer, Ryan; Buchanan, Erik; Shacham, Hovav; Savage, Stefan: Return-Oriented Programming: Systems, Languages, and Applications. ACM Trans. Inf. Syst. Secur., 15(1):2:1–2:34, March 2012.

[SAIC08] Science Applications International Corporation, Common Criteria Testing Laboratory: , INTEGRITY-178B Separation Kernel Security Target Version 1.0, 2008.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 229

On the Security of the ZigBee Light Link Touchlink Commissioning Procedure

Frederik Armknecht[1], Zinaida Benenson[2], Philipp Morgner[3], Christian Müller[4]

Abstract: Specifications of security mechanisms often lack explicit descriptions of the envisioned security goals and the underlying assumptions. This makes it difficult for developers and customers to understand the level of security provided by the systems. Moreover, this omission has repeatedly resulted in practical attacks that violate the implicit security assumptions of the specifications. In this work, we illustrate this effect on the example of the ZigBee Light Link (ZLL) profile, currently one of the most popular standards for smart lighting in domestic environments. We first provide a concise description of ZLL commissioning procedure for initiating and extending a network of smart bulbs, extracted directly from the specification. We then discuss how the commissioning protocol can be transformed into a formal security model, but also highlight where this is subject to interpretations because of the unclear implicit security assumptions. The proposed security model is flexible, i.e., it can be extended to capture further security requirements or attacker classes, and hence provides a solid foundation for rigorous security analyses of ZLL and other ZigBee profiles.

Keywords: Smart Home, ZigBee Light Link, Security Model.

1 Introduction

Smart buildings comprise interconnected household, office and personal devices, and systems designated to support and enhance the everyday life of inhabitants. Applications include alarm systems, energy management, building automation, or support systems for elderly and disabled people [BLM+11, HH12, MH12]. These systems can have a substantial impact on safety and privacy of the users, as they collect and process sensor data about the residents, and also automatically trigger different actuators.

Therefore, security mechanisms should ensure that only authorized entities control and manage smart objects, as well as that only authorized smart objects communicate with each other and access the gathered information [UJS13, ARA12]. Despite the enormous relevance of this topic, a comprehensive framework for a sound analysis of security features in smart building networks does not exist. Even worse, for most existing mechanisms in place, a rigorous description of the underlying security assumptions is missing.

This is not of academic interest only. On the contrary, security holes discovered in practice are often difficult to fix afterwards and mostly result in high costs, as the security disasters of GSM [BBK03], WEP [BHL06] and Bluetooth [JW01, HHPT13] have shown. Quite often in these cases, the discovered vulnerabilities were either apparent in the specifications from the very beginning, or resulted from the inaccurate or missing assumptions

[1] University of Mannheim, armknecht@uni-mannheim.de
[2] University of Erlangen-Nuremberg, zinaida.benenson@fau.de
[3] University of Erlangen-Nuremberg, philipp.morgner@fau.de
[4] University of Mannheim, christian.mueller@uni-mannheim.de

about what to protect (security goals) and against whom to protect (attacker motives and capabilities).

Security mechanisms of ZigBee Pro [Zig12b], one of the most popular standards for low-power and low-cost networks in smart buildings, similarly have turned out to be insufficient over and over again [Wri09, VHP+13, Zil15]. This creates the impression of smart buildings being insecure in general, which may unsettle potential customers. Therefore, it is necessary to provide a concise and well-founded assessment of ZigBee security.

In this work, we make a first step towards this goal and introduce a formal security model for the touchlink commissioning procedure described in the ZigBee Light Link (ZLL) [Zig12a] profile specification. We demonstrate how to express ZLL security mechanisms by means of a formal security model, at the same time discussing the shortcomings of the corresponding specification where the readers have to rely on personal interpretations. The introduced security model can be easily extended for analysis of further security goals and mechanisms in smart buildings and similar systems.

This paper is organized as follows: In Section 2, we outline related work. We provide background on ZigBee and ZLL in Section 3, and describe ZLL touchlink commissioning in Section 4. In Section 5, we present the formal security model for ZLL and discuss the security of ZLL within this model. We conclude this paper in Section 6.

2 Related Work

To the best of our knowledge, neither a formal concept of a security model nor a sound security framework for the ZLL standard or parts thereof has been published yet. However, several works analyze security of ZigBee products and outline security flaws. Thus, Wright [Wri09] introduced the KillerBee framework as a penetration testing tool for ZigBee networks. He exposed security flaws in the key provisioning protocol of the ZigBee Pro specification by showing that the network key is sent unprotected on the wireless channel. Vidgren et. al. [VHP+13] demonstrated the exploitation of the key exchange process in the Standard Security mode of ZigBee. As a consequence to this attack, they conclude that the Standard Security mode should be removed from the ZigBee specification.

Zillner and Strobl [Zil15, ZS15] analyzed security measures of the ZigBee Pro Home Automation (ZHA) and the ZLL profiles using their SecBee framework and found numerous protocol and implementation flaws. For example, they exploit the ZLL touchlink commissioning process to seize control over ZLL devices and show that unauthorized disclosure of the ZLL master key in March 2015 compromised the security of the entire ZLL ecosystem.

3 ZigBee Light Link

In this section, we introduce ZLL and provide an overview of the technical background and security mechanisms.

3.1 ZigBee Overview

ZigBee is a standard that connects embedded technologies in Wireless Personal Area Networks (WPANs). The specification is maintained by the ZigBee Alliance, a non-profit organization that comprises over 250 members. It defines the network, security, and software layers and supervises the conformance and interoperability of ZigBee-certified products.

The first ratified ZigBee specification was published in 2004, referred to as *ZigBee 2004*. Two years later, in the *ZigBee 2006* specification, cluster libraries and security features were introduced. A new security model including a so-called *trust center* was introduced in the *ZigBee 2007* specification, and in the same year, the initial *ZigBee Pro* specification containing additional software features and enhanced security mechanisms was released, which has been subject to several revisions, the most recent in 2012. The newest *ZigBee 3.0* standard is expected to be published in 2016 and is not publicly available yet. In this paper, we thus consider the current *ZigBee Pro* specification [Zig12b].

ZigBee Pro includes different *profiles*, which comprise customized sets of features and protocols for a specific application area, such as Home Automation (ZHA) [Zig13b] and Smart Energy (SE) [Zig13a] profiles. Here we focus on the ZLL profile [Zig12a] that is implemented, for example, by Philips Hue and Osram Lightify are smart lighting systems.

3.2 ZLL Technical Background and Security Mechanisms

The ZLL profile describes three different node types: ZigBee Coordinator (ZC), ZigBee Router (ZR), and ZigBee End Device (ZED). In a ZigBee network, there exists exactly one designated ZC, which acts as the router initially forming the network. In contrast, there is no fixed number of ZRs and ZEDs. ZRs act as full-function devices that route data frames to other nodes, whereas ZEDs do not have router functionality and need a ZC or ZR parent device for network participation. Most ZigBee devices form mesh networks, but also other network topologies are feasible, such as star and tree.

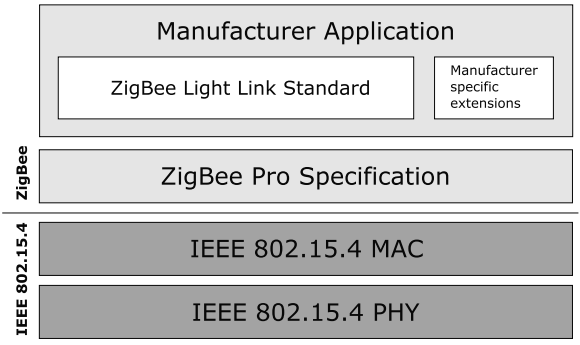


Fig. 1: Overview of the stack architecture of ZLL devices.

As illustrated in Figure 1, the ZLL stack consists of four layers: physical (PHY), medium access control (MAC), network (NWK), and application (APL). The two lower layers,

PHY and MAC, are defined in the IEEE 802.15.4-2003 specification [IEE03]. This specification is also incorporated into other WPAN standards, most prominently Thread Group [Sil15] and WirelessHART [Int10]. The PHY layer is the lowest layer and defines the physical interface. ZigBee products use radio-frequency communication in the 2.4 GHz ISM band. The MAC layer provides functionalities to transmit data frames in a reliable manner by managing access to the radio channel via CSMA/CA mechanisms, sending beacon frames, acknowledgement frames, and applying synchronization mechanisms.

The network layer is specified in the ZigBee Pro standard and manages network topology, routing, parts of the security services, and acts as a message broker. The application layer contains the manufacturer applications. This layer also includes the application profiles, which for ZLL products is the ZLL standard. Since the touchlink commissioning procedure is defined in the ZLL standard, this procedure is also defined in this layer.

Security measures are only applied to the two upper layers, NWK and APL. The two lower layers also support encryption and integrity mechanisms, however, these are not supported in ZigBee products. ZLL devices use the AES-CCM* authenticated encryption scheme with a 128-bit key. ZLL devices hold three types of cryptographic keys: link keys, the network key, and the ZLL master key. A link key is only shared between two devices in a network and used for direct unicast communication. Consequently, a ZLL device may hold multiple link keys. The network key is used for broadcast communication and shared between all devices of a network. The ZLL master key is shared between all devices that support the ZLL profile and is distributed to manufacturers by the ZigBee Alliance and protected by a non-disclosure agreement (NDA). Nonetheless, the ZLL master key was disclosed in March 2015 as stated by Zillner [Zil15].

4 ZLL Touchlink Commissioning

For the initial setup, a device needs to be associated to the customer's WPAN. The procedure of connecting a new device to a network for the first time is called commissioning. ZLL enables two commissioning modes: touchlink commissioning and classical commissioning. Since we focus on the touchlink commissioning, the classical commissioning is not further considered in this paper.

4.1 Protocol Description

We explain the commissioning protocol for joining a new device to an existing network in an abstract manner (see Figure 2), omitting the details that do not pose any additional information concerning the scope of this paper.

The touchlink commissioning protocol is executed between two entities, an initiator and an end device. The initiator is usually either a controller or a bridge, i. e., a ZLL device characterized by a physical button which is pushed to start the commissioning procedure. The end device represents the ZLL device to be added to the network. The initiator starts

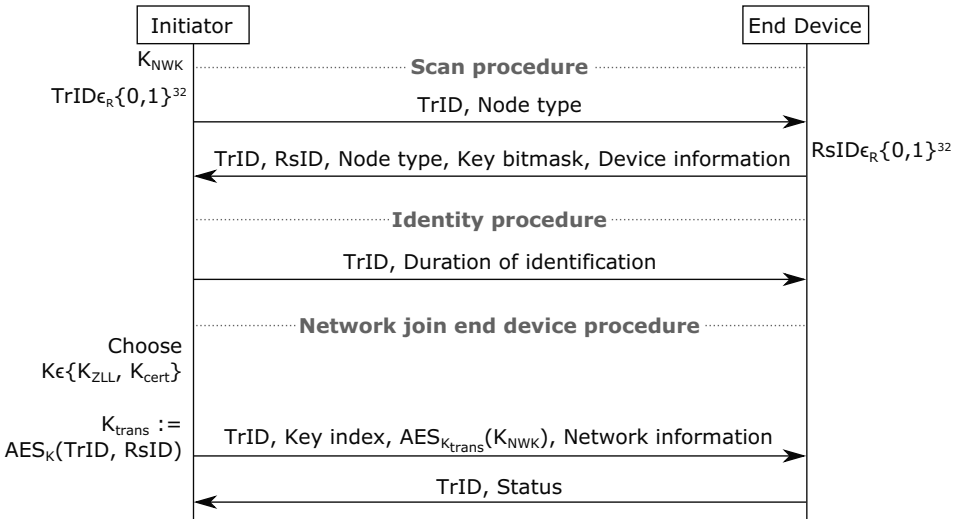


Fig. 2: Touchlink commissioning protocol in the ZLL profile using either the ZLL master key K_{ZLL} or the certification key K_{cert} to encrypt the network key K_{NWK}. `TrID` and `RsID` denote the random 32-bit transaction and response identifier, respectively. The notation $x \in_R \{0,1\}^l$ denotes a randomly chosen bit string x of length l.

the device scan procedure by sending *scan requests* on different channels as defined in the ZLL specification. These scan requests include the randomly generated transaction identifier `TrID` and the node type of the initiator (cf. Section 3.2). The end device replies with a *scan response* containing `TrID`, a random response identifier `RsID`, its own node type, the list of all supported keys referred to as `key bitmask`, and some further information.

The device scan may yield multiple potential devices from which the user can select at least one for the next steps. The user has the option to send an *identity request* to a device, upon which the target device performs a predefined identification action, e.g., a light bulb will flash a few times. An *identity request* contains the corresponding `TrID` as well as the duration of the identification action.

To join a new end device to a network, the initiator encrypts the current network key using as outlined in Section 4.2, builds a *network join end device request* containing this encrypted network key, `TrID`, the key index as well as network information, and then sends this request to the selected end device. On receiving the message, the joining end device replies with a *network join end device response* which includes a status flag set to indicate success.

4.2 Network Key Encryption

The *scan response* contains a key bitmask, which indicates the keys available to the end device. These can be the *ZLL master key*, the *Certification key*, or the *Development key*.

The initiator compares the received key bitmask to its own keying material and chooses the appropriate key. As the Development key uses a slightly different encryption algorithms and is only allowed during the development phase, here we concentrate on the algorithm that is utilized with the ZLL master key and the Certification key.

The ZLL master key, denoted K_{ZLL}, is distributed to certified manufacturers of ZLL products and bound with a non-disclosure agreement (NDA). The Certification key is used to evaluate the security mechanisms during the certification phase of a ZLL product and is defined in the ZLL specification.

To encrypt the randomly chosen network key, the initiator expands the transaction identifier and the response identifier to a 128-bit number which represents the plaintext input to the AES-ECB encryption, while either the ZLL master key or the Certification key is used as encryption key. The resulting ciphertext output is called Transport key. In the next step, the actual network key, denoted K_{NWK} is encrypted under the Transport key, again using the AES-ECB encryption mode.

We note that AES-ECB does not support any integrity protection, which makes the following Denial of Service attack theoretically possible: The attacker disturbs the part of the message containing encrypted K_{NWK} in such a way that some bits in the AES-ECB encryption are flipped, and everything else remains unaffected. In this case, the End Device will accept the message and decrypt the supposed K_{NWK} into garbage without noticing.

5 Security Model

5.1 The Contradictory ZLL Security Assumptions

As discussed in Section 1, a concise security model is indispensable for a meaningful security analysis. Such a security model has to specify at least the following three components:

1. *System model* describes what entities and possible actions are considered, such as types of devices and communication between them.
2. *Security goals* specify what should be protected, e.g., confidentiality and integrity of data.
3. *Attacker model* establishes against which attack types security is envisioned. For example, it describes the capabilities of the attackers, such as eavesdropping or physical device compromise.

Indeed, without knowing the attackers' capabilities and the security goals of a system, it is impossible to claim that a system is secure or insecure, because one does not know what should be protected (goals) and against which threats (attackers).

The ZLL system model is rigorously described in the 120 pages of the corresponding specification [Zig12a]. Also the functionality of the ZLL Touchlink Commissioning protocol

and other security protocols is precisely specified, including the format and the order of all protocols messages. Unfortunately, this document does not say anything about the attacker model and the security goals, apart form stating that *"The ZLL master key is a secret shared by all certified ZLL devices. It will be distributed only to certified manufacturers and is bound with a safekeeping contract"* [Zig12a, page 104].

The reason behind this approach is quite understandable: whereas message formats and order of protocol messages are crucial for the operation of the network (they are *functional* requirements), attacker model and security goals are *non-functional* requirements: if the attacker is not present, the ZLL network is going to work as long as all specified mechanisms are implemented correctly.

Turning to the ZigBee specification [Zig12b], we find the "Security Assumptions" section on pages 426-427. This is the only place in this 600 pages long document where the security model is considered. The description of security assumption is quite informal. Especially relevant for our work is the following statement: *"[...] due to the low-cost nature of ad hoc network devices, one cannot generally assume the availability of tamper-resistant hardware. Hence, physical access to a device may yield access to secret keying material and other privileged information, as well as access to the security software and hardware."*

Thus, when trying to extract the attacker model and the security goals from the specifications, we discover contradictory statements that make the derivation of a precise formal security modeling impossible. On the one hand, ZigBee specification envisions the attacker that is able to gain access to the internal secrets of the devices by physical tampering. On the other hand, ZLL specification builds the security of the entire ZLL ecosystem on the single master key that is the same for all ZLL devices.

In the following, we nevertheless attempt to model ZLL security more precisely and show possible pitfalls. We start with presenting a system model, then discuss (some) security goals, and finally address the attacker model.

5.2 System Model

Within the formal system model, one has to specify the involved entities and the actions they can take. Here, we exclude the attacker as this is part of the attacker model presented in Section 5.4. On a high level, the system comprises devices that can communicate with each other and also jointly execute protocols.

Nodes. The entities of the system are given by a set of all ZLL nodes $\mathcal{N}$.[5] This set is not fixed but may vary over time, i.e., nodes may be added or removed. Among the set of nodes, there is a particular class of nodes called the *coordinator nodes*.

In cryptographic models, one usually assumes that the set of all nodes is fixed. In contrast, we assume a process Create that has no input and outputs a new node. That is, we express

[5] We use the term *node* instead of ZigBee device to be consistent with other models (cf. [GRT12, VJU+12]).

by $\mathsf{n} \leftarrow \mathsf{Create}$ the event that Create has been initiated and its output is a new node n. In this case, the set of all nodes is extended by the new node n, that is, $\mathscr{N}$ is updated to $\mathscr{N} \cup \{\mathsf{n}\}$. Nodes that have been created by Create (and only those) are called *genuine* nodes and we refer by $\mathscr{N}_{\text{gen}}$ to the set of all genuine nodes.

We do not further specify Create, as its real-world meaning can depend on the concrete use case. For example, Create may refer to the process of getting (buying) nodes from a certain vendor only, or may comprise nodes that have been approved by a certain authority.

Protocols. Nodes can communicate with each other and also jointly run protocols. To express the situation that two nodes $\mathsf{n1}$ and $\mathsf{n2}$ jointly run a protocol Π, we adopt the following common notation from cryptography:

$$\Pi : [\mathsf{n1} : x_1, \mathsf{n2} : x_2] \rightarrow [\mathsf{n1} : y_1, \mathsf{n2} : y_2] \tag{1}$$

This expresses that both nodes run the protocol Π where x_1 and x_2 denote the inputs of $\mathsf{n1}$ and $\mathsf{n2}$, respectively, and y_1 and y_2 the respective outcomes of the protocol. Here, "input" and "outcome" refer to locally known values only. For example, x_1 is the value used by $\mathsf{n1}$ to run the protocol but does not necessarily mean that it is sent to $\mathsf{n2}$. We use $\perp$ to indicate an uninitialized parameter as well as an intended lack of input or output, e. g., $y_2 = \perp$ would refer to the case that $\mathsf{n2}$ has no internal output of the protocol.

In our system model, we cover two protocols: Π_{join} and Π_{cmd}. In a nutshell, the first protocol coordinates the inclusion of nodes into the network, while the second allows nodes within a network to exchange commands. Note that a full system model would require to specify the *correctness* of protocols. We omit this here due to space limitations. For the same reason, we discuss the protocols here on an abstract level only and shortly address later how these are realized in ZLL. The Π_{join} protocol is executed between a node n and a coordinator c and has the following specification:

$$\Pi_{\text{join}} : \left[\mathsf{c} : x, \mathsf{n} : x'\right] \rightarrow [\mathsf{c} : \delta, \mathsf{n} : y] \tag{2}$$

where x, x' are the inputs of c and n, y is the local outcome of n and $\delta \in \{\text{accept}, \text{reject}\}$ denotes whether the protocol run is successful or not.Similarly, the Π_{cmd} protocol is executed between two nodes $\mathsf{n1}, \mathsf{n2}$:

$$\Pi_{\text{cmd}} : \left[\mathsf{n1} : x, \mathsf{n2} : x'\right] \rightarrow [\mathsf{n1} : \perp, \mathsf{n2} : \delta] \tag{3}$$

Again, x, x' denote the inputs of the nodes and $\delta \in \{\text{accept}, \text{reject}\}$ denotes whether the protocol run is successful or not.

Networks. As already mentioned, nodes can be grouped into *networks*. Each network has exactly one coordinator node c. We use this fact to formally define a network:

Definition 1 (Network)**.** *Consider the set of all nodes $\mathscr{N}$. A subset $\mathscr{N}_{\mathsf{c}} \subseteq \mathscr{N}$ is called a* network *with respect to some coordinator node c if for any $\mathsf{n1} \in \mathscr{N}_{\mathsf{c}}$ and $\mathsf{n2} \in \mathscr{N}_{\mathsf{c}}$ the following holds: If $\mathsf{n1}$ runs the Π_{cmd} protocol with $\mathsf{n2}$, then $\mathsf{n2}$ eventually accepts.*

5.3 Security Goals

In the following, we specify the security goals of ZLL. They are not given in the specification and hence are based on our interpretation. According to the specification, each node is equipped with a master key K_{ZLL} which is used in the Π_{join} protocol to transport a network key K_{NWK} from the coordinator to the node. The knowledge of the network key K_{NWK} is mandatory to make the other node accept in the Π_{cmd} protocol. Based on these observations, our interpretation of the security goals are:

1. Only nodes that know the master key K_{ZLL} should be allowed to join a network.
2. Only nodes that know the network key K_{NWK} should be allowed to send commands to other nodes in the network.

As the knowledge of K_{ZLL} is the only condition for being able to join the network, we make the following interpretation: the set of genuine nodes $\mathcal{N}_{\text{gen}}$ is, according to the specification, nodes that know K_{ZLL}.[6] Likewise, we can say that in ZLL, the set of nodes $\mathcal{N}_{\mathsf{c}}$ that belong to the same network are characterized by sharing the same network key K_{NWK}.

Based on these considerations, we propose the following two security definitions:

Definition 2 (Security of Π_{join})**.** *Π_{join} is $\varepsilon_{\text{join}}$-secure if the probability that Π_{join} between nodes* c *and* n *is successful does not exceed $\varepsilon_{\text{join}}$ for any $\mathsf{n} \notin \mathcal{N}_{gen}$.*

Definition 3 (Security of Π_{cmd})**.** *Π_{cmd} is ε_{act}-secure if for any network $\mathcal{N}_{\mathsf{c}}$ the following holds: for any $\mathsf{n1} \notin \mathcal{N}_{\mathsf{c}}$ and $\mathsf{n2} \in \mathcal{N}_{\mathsf{c}}$ the success probability of Π_{cmd} between* $\mathsf{n1}$ *and* $\mathsf{n2}$ *does not exceed ε_{act}.*

These security goals are met in the system if both security definitions hold for small values $\varepsilon_{\text{join}}$ and ε_{act}. We note that further security requirements can be identified, such as data confidentiality, but again this has to be omitted due to lack of space.

5.4 Attacker Model

In the next step, we formalize the attacker. Different attacker types are imaginable here. To keep the model as flexible as possible, we follow the common approach in cryptography and formalize the attacker's capabilities by so-called queries. That is, formally an action of the attacker is captured by saying that the attacker makes the corresponding query (e.g., to a hypothetical party which then executes this action on behalf of the attacker). Obviously, the more types of queries an attacker can make, the more powerful she is.

The first two queries express the capability of an attacker to control the communication between nodes:
RECEIVE(n) – Receive all messages sent to node n (including broadcasts).

[6] More generally, one could say that there is an external procedure which guarantees that only genuine nodes known K_{ZLL}. But this would be out of scope of ZLL and hence is not considered here.

SEND(n, m) – Send message m to node n.
In principle, this would be sufficient to for two nodes to execute a certain protocol. However, protocol runs may involve secrets that are unknown to the adversary. On the other hand, it may be possible to remotely trigger certain protocols, e.g., by touching a button on a device. This is captured by the following query:
RUN($\Pi, \mathsf{n1}, \mathsf{n2}$) – triggers protocol Π between the two nodes $\mathsf{n1}$ and $\mathsf{n2}$.

An adversary may also be able to introduce her own nodes to the system or remove nodes. Here, we distinguish between genuine nodes (which result from the Create process) and nodes created by the adversary:
ADDGENUINE(n) – adds a genuine node n to $\mathscr{N}$, i. e., a node created by Create.
ADDCUSTOM(n) – adds a node n to $\mathscr{N}$, which is under the full control of the adversary, meaning that she knows the complete state of the node including any secrets.
REMOVE(n) – the node n is removed completely from the system, e.g., by destroying it, removing the power source, etc.

Finally, an adversary may bring a (possibly genuine) node under her full control, meaning that she learns all internal values and can reprogram the node at her wish. We denote the corresponding query by CORRUPT(n).

5.5 Discussion

We already discussed that in ZLL, the Π_{join} protocol actually only checks if the node knows the master key K_{ZLL}, meaning that in reality, genuine nodes are characterized by knowing this value. Likewise, members of a network are characterized by sharing the same network key K_{NWK}. Observe that the Π_{join} protocol essentially established the network key between two nodes by sending its encryption using K_{ZLL} as the encryption key. Moreover, in Π_{cmd}, the commands are sent encrypted under the encryption key K_{NWK}. Thus, one can show[7] that if the encryption scheme is secure and both keys are secret, then both security definitions are fulfilled.

As the ZigBee specification admits that nodes may be physically attacked, the ZLL master key is subject to unauthorized disclosure. After this actually happened in March 2015 [Zil15], virtually any device is now able to pretend being a genuine node, violating the security definition given in Def. 2. Similarly, the network key K_{NWK} may be leaked, allowing a node to execute the Π_{cmd} protocol without having run Π_{join} before.

However, as we already stated, the choice of security definitions are also subject to interpretations. One could likewise define a network to be the set of all nodes that successfully executed the Π_{join} protocol with c. This definition would be violated if K_{NWK} is leaked. Still, we don't see this as the "correct" definition as the purpose of the Π_{join} protocol should be to achieve a given, independent security goal and not be the basis of a security definition. In any case, this again shows the necessity of giving concise security definitions. Moreover, observe that the security goals are violated only in presence of a physical

[7] In fact, one can conduct a full formal security proof if one assumes a secure encryption scheme. But again, this has to be omitted due to lack of space.

attacker, i.e., an attacker who can submit CORRUPT(n) queries. This demonstrates the importance of providing a clear attacker model.

6 Conclusion

In this paper, we provided technical background of the ZLL profile with focus on security mechanisms, especially on those used in the commissioning process, which is crucial to enable successful communication among ZLL nodes. Based on this, we developed a formal security model specifically for ZLL and reviewed the security mechanisms within this model. While this represents a necessary step towards a sound security analysis of ZLL, several fundamental security goals such as confidentiality and availability are not covered yet and are thus left for future research.

Acknowledgments This research was supported by the DFG project "Developing and Applying a Sound Security Framework for Sensor Networks".

References

[ARA12] Muhammad Raisul Alam, Mamun Bin Ibne Reaz, and Mohd. Alauddin Mohd. Ali. A Review of Smart Homes - Past, Present, and Future. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 42(6):1190–1203, 2012.

[BBK03] Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. In *Advances in Cryptology-CRYPTO 2003*, pages 600–616. Springer, 2003.

[BHL06] Andrea Bittau, Mark Handley, and Joshua Lackey. The final nail in WEP's coffin. In *Security and Privacy, 2006 IEEE Symposium on*, pages 15–pp. IEEE, 2006.

[BLM+11] A. J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon. Home automation in the wild: challenges and opportunities. In Desney S. Tan, Saleema Amershi, Bo Begole, Wendy A. Kellogg, and Manas Tungare, editors, *Proceedings of the International Conference on Human Factors in Computing Systems, CHI 2011, Vancouver, BC, Canada, May 7-12, 2011*, pages 2115–2124. ACM, 2011.

[GRT12] Laura Gheorghe, Razvan Rughinis, and Nicolae Tapus. Adaptive Security Framework for Wireless Sensor Networks. In Fatos Xhafa, Leonard Barolli, Florin Pop, Xiaofeng Chen, and Valentin Cristea, editors, *2012 Fourth International Conference on Intelligent Networking and Collaborative Systems, INCoS 2012, Bucharest, Romania, September 19-21, 2012*, pages 636–641. IEEE, 2012.

[HH12] Amy S. Hwang and Jesse Hoey. Smart Home, The Next Generation: Closing the Gap between Users and Technology. In *Artificial Intelligence for Gerontechnology, Papers from the 2012 AAAI Fall Symposium, Arlington, Virginia, USA, November 2-4, 2012*, volume FS-12-01 of *AAAI Technical Report*. AAAI, 2012.

[HHPT13] Keijo Haataja, Konstantin Hyppönen, Sanna Pasanen, and Pekka Toivanen. *Bluetooth Security Attacks: Comparative Analysis, Attacks, and Countermeasures*. Springer Science & Business Media, 2013.

[IEE03] IEEE Computer Society. IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks Specific Requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std 802.15.4-2003*, pages 1–670, 2003.

[Int10] International Electrotechnical Commission. *IEC 62591 Ed. 1.0: Industrial communication networks – Wireless communication network and communication profiles – WirelessHART*, 2010.

[JW01] Markus Jakobsson and Susanne Wetzel. Security weaknesses in Bluetooth. In *Topics in Cryptology – CT-RSA 2001*, pages 176–191. Springer, 2001.

[MH12] Sarah Mennicken and Elaine M. Huang. Hacking the Natural Habitat: An In-the-Wild Study of Smart Homes, Their Development, and the People Who Live in Them. In Judy Kay, Paul Lukowicz, Hideyuki Tokuda, Patrick Olivier, and Antonio Krüger, editors, *Pervasive Computing - 10th International Conference, Pervasive 2012, Newcastle, UK, June 18-22, 2012. Proceedings*, volume 7319 of *Lecture Notes in Computer Science*, pages 143–160. Springer, 2012.

[Sil15] Silicon Labs. *Application Development Fundamentals: Thread, Rev. 0.3*, 2015.

[UJS13] Blase Ur, Jaeyeon Jung, and Stuart Schechter. The current state of access control for smart devices in homes. In *Workshop on Home Usable Privacy and Security (HUPS)*, 2013.

[VHP+13] Niko Vidgren, Keijo Haataja, Jose Luis Patino-Andres, Juan Jose Ramirez-Sanchis, and Pekka Toivanen. Security Threats in ZigBee-Enabled Systems: Vulnerability Evaluation, Practical Experiments, Countermeasures, and Lessons Learned. In *46th Hawaii International Conference on System Sciences, HICSS 2013, Wailea, HI, USA, January 7-10, 2013*, pages 5132–5138. IEEE, 2013.

[VJU+12] Marco Valero, Sang Shin Jung, A. Selcuk Uluagac, Yingshu Li, and Raheem A. Beyah. Di-Sec: A distributed security framework for heterogeneous Wireless Sensor Networks. In Albert G. Greenberg and Kazem Sohraby, editors, *Proceedings of the IEEE INFOCOM 2012, Orlando, FL, USA, March 25-30, 2012*, pages 585–593. IEEE, 2012.

[Wri09] Joshua Wright. KillerBee: Practical ZigBee Exploitation Framework. ToorCon 11, 2009.

[Zig12a] ZigBee Alliance. *ZigBee Light Link Standard Version 1.0 – Document 11-0037-10*, April 2012.

[Zig12b] ZigBee Standards Organization. *ZigBee Specification – Document 053474r20*, September 2012.

[Zig13a] ZigBee Alliance. *Smart Energy Profile 2 Application Protocol Standard – Document 13-0200-00*, April 2013.

[Zig13b] ZigBee Alliance. *ZigBee Home Automation Public Application Profile Version 1.2 – Document 05-3520-29*, June 2013.

[Zil15] Tobias Zillner. White paper: ZigBee Exploited – The good, the bad and the ugly. Technical report, Cognosec, August 2015.

[ZS15] Tobias Zillner and Sebastian Strobl. ZigBee Exploited – The good, the bad and the ugly. Black Hat USA, 2015.

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 241

Increasing security and availability in KNX networks

Harald Glanzer[1], Lukas Krammer[1], Wolfgang Kastner[1]

Abstract: Buildings contain a number of technical systems in order to be able to fulfill their task of providing a comfortable, secure and safe environment. Apart from heating, ventilation and air-conditioning as well as lighting and shading, critical services such as fire alarm or access control systems are added to building automation. The latter services require secure communication and high availability and are currently implemented by isolated subsystems. However, a tighter integration into an overall building automation network can raise synergies such as cost reduction, improvements in building control as well as easier management. For this purpose, the underlying communication system has to be robust and reliable against malicious manipulations. This paper proposes an extension for KNX paving the way for its deployment even in critical environments. For this purpose, it is necessary to detect and guard against malicious attacks as well as to cope with randomly occurring hardware faults. The former can be achieved through cryptography, whereas the latter by implementing structural redundancy. The proposal divides KNX installations into insecure and secure parts. While insecure parts allow to use standard KNX devices, secure parts are protected against malicious attacks and are realized in a redundant way. This allows to partially resist against transient hardware faults.

Keywords: Building automation, Security, Availability, KNX, Smart building

1 Introduction

Building automation (BA) is a widespread topic that evolved over the past decades [Ka05]. Initially, BA systems were used for heating, ventilation and air-conditioning (HVAC) applications as well as for lighting and shading. Today, the term BA covers many more application domains such as alarm systems, access control or safety systems. Combining all these domains leads to ”intelligent buildings” and promises reduced maintenance costs, energy savings and improved user comfort compensating the primarily higher investment costs of such buildings. However, each application domain has different demands regarding dependability with the attributes reliability, availability, safety, confidentiality, integrity and maintainability [Av04].

In BA, the exchange of control data is a key issue. Small amounts of data are transmitted infrequently, but dependably over long distances. A number of control networks cater for this domain (e.g., BACnet, LonWorks, KNX, ZigBee, DALI). Integrating all application fields of BA mentioned above into an overall control network would, nevertheless, raise synergies in terms of sensor fusion and sensor sharing, as long as such a system could meet all dependability attributes.

In the early days of BA, this vision was contradicted by the fact that (communication) security was not considered as a critical requirement. The possible threats by misusing

[1] Technische Universität Wien, Automation Systems Group, contact author: k@auto.tuwien.ac.at

HVAC applications were assumed to be negligible. Additionally it was argued that the control networks were physically isolated. Finally, the underlying nodes were characterized by very limited processing power. Thus, the comprehensive use of cryptographic mechanisms would have put remarkable computing loads onto these nodes and was considered impracticable.

Meanwhile, system integration continued until a point where security concerns can no longer be neglected. BA systems and their networks are more and more linked together crossing former building borders for opening new application fields (e.g., for demand and load side management in smart grids). Considering such applications, a wide range of attacks has become possible. Intercepting and replaying datagrams allows an adversary to introduce arbitrary control data, for instance, to open doors or to disable HVAC systems without permission. Passive attackers can monitor the network traffic to analyze the types of active devices, gathering knowledge that can be used to develop further attack strategies. Denial-of-Service (DoS) attacks disabling building services can be conducted by simply physically shortcutting or interrupting a line connection, rendering the corresponding network segment unavailable. Such attacks must be precluded for sensitive services like fire or burglar alarm systems relying on the availability of the communication network. Availability, in general, can only be achieved by structural redundancy, i.e., by using replicated resources. Therefore, all resources needed for transmitting data between two points must exist redundantly and independently.

KNX is an open and widespread BA technology. It uses a layered structure and supports wired communication over Twisted Pair (TP) and Power Line (PL) as well as wireless communication by radio transmission. In addition, it supports communication with IP based hosts by a special type of router (KNXnet/IP). The present paper is focused on the design of a secure and highly available KNX network that also considers interoperability and backward compatibility, allowing the usage of KNX even in environments with increased safety-critical requirements. The proposed solution claims to be resistant against malicious adversaries as well as transient and permanent hardware faults. To achieve this, so called ”KNX security routers” are introduced. These devices possess two kinds of KNX interfaces. One kind of interface is connected to standard KNX networks. The second interface constitutes the entry point to a secured KNX network which is connected to the secured interfaces of other KNX security routers. To achieve higher availability, these secured interfaces and the corresponding communication lines must exist redundantly. This ensures that even in case of a DoS attack against one physical connection, communication is still possible.

2 Building automation networks and attack scenarios

Communication networks for BA are usually built upon a two-tier model consisting of a field level and a backbone level [Ka05]. The field level contains sensors, actuators and controllers interacting with the environment and performing the control functions. The field level devices are connected via fieldbus systems which in turn are coupled to a common backbone network via interconnection devices. The backbone network is home for man-

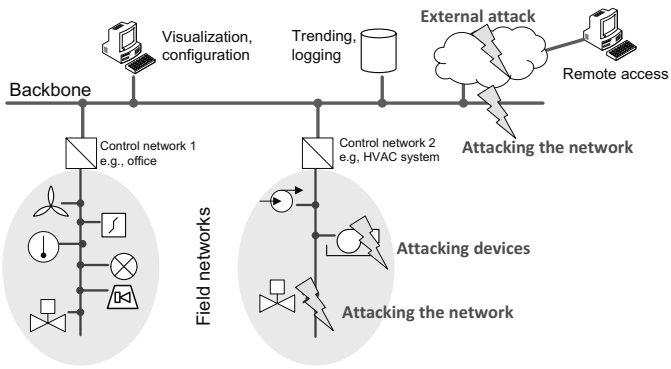


Fig. 1: Building automation networks and attack scenarios

agement devices which are used for configuration, visualization and monitoring. Special interconnection devices may act as gateways providing a connection to foreign networks (Figure 1).

Based on this topology, two different classes of attacks are possible: network attacks and device attacks [Gr10]. In case of network attacks, an adversary tries to compromise the field or backbone network. In the first scenario, the attacker can analyze or modify the control data of specific network segments. In the latter, the attacker gains access to the aggregated data and can thus obtain a global view of the system. To protect the communication, it would be possible to use security mechanisms well-known from the IT world. Unfortunately, these mechanisms cannot be mapped directly to BA networks because of the introduced overhead.

Alternatively, attacks against sensors, actuators or controllers to manipulate their behavior or attacks against interconnection devices to limit data access to and from the specific segments are possible. Finally, an adversary can launch an attack against the management devices to gain control over management applications (e.g., for accessing trends or logs). In general, such device attacks are divided into three categories: software attacks, side-channel and physical attacks, extensively surveyed in [GPK10] and [KS05].

3 KNX and KNX Security

KNX emerged from three leading standards namely European Installation Bus (EIB), European Home Systems Protocol (EHS) and Batibus. It is an open, platform independent standard, developed by the KNX Association implementing the ISO/IEC 14543-3 standard for home and building electronic systems. To provide platform independence, the standard uses a layered structure, based on the OSI model omitting layers 5 and 6. Different kinds of physical media are supported, allowing its use in different environments. Two different kinds of addresses are defined. The Individual Address (IA) of a KNX node is related to its position within the topological structure of the network. It specifies the number of the zone and (sub-)line the device resides in as well as its device number within

the line. Group Addresses (GA) are logical identifiers and used for group communication. KNX supports interoperability between products from different manufacturers. This is achieved by an interworking model, which rests upon the concept of functional blocks and standardized data point types. For configuration and parametrization of the devices, the Engineering Tool Software (ETS) is used.

KNX does not implement security features except a rudimentary password-based control for management communication. The used keys are transmitted as cleartext, enabling an attacker to perform a passive attack to obtain the password. Subsequently, the attacker can mount an active attack, injecting arbitrary management messages. No methods are foreseen for generation or distribution of the keys. For control data, an adversary can directly inject arbitrary messages. These shortcomings disqualified KNX for usage in critical environments, restricting its possible fields of application. For this reason, a number of security extensions has been proposed (e.g., [Le09, CCM10]).

3.1 KNX Data Security

In 2013, the KNX Association published ”KNX Application Note 158” which specifies the KNX Secure Application Layer (S-AL) providing authentication and encryption, and the Application Interface Layer (AIL) implementing access control, both being part of the application layer. The settlement of these functions above the transport layer allows a transparent and media-independent end-to-end protection comprising of encryption and authentication [Kr13].

KNX S-AL services define modes for authenticated encryption or authentication-only of a higher-level cleartext Application Protocol Data Unit (APDU). This is achieved by combining Counter Mode AES-128 for encryption with Cipher Block Chaining (CBC) for generating the Message Authentication Code (MAC), also known as Counter with CBC-MAC (CCM). A critical parameter of CCM is the sequence number, a simple counter value that provides data freshness, thus preventing replay attacks, and is sent along with every KNX S-AL protocol data unit. This sequence number has to be synchronized by communicating devices. Since no sequence number can be used to guarantee data freshness at this stage, a challenge-response mechanism is used instead. For encrypting and authenticating KNX frames, different types of keys are used: a Factory Default Setup Key (FDSK) is used for initial setup with the ETS. The ETS then generates the Tool Key (TK) which is used by the device for securing of the outgoing messages. Consequently, every device must know the TK of its communication partners. While the S-AL empowers two devices to communicate in a secure way, the AIL allows a fine-grained control access control. Therefore, every link (a combination of source address and data or service object) is connected with a role, which in turn has some specific permissions.

3.2 KNX IP Secure

KNX IP Secure, as published in ”KNX Application Note 159”, is a security extension for KNXnet/IP that aims at being backward compatible. The KNXnet/IP traffic is encapsu-

lated in KNXnet/IP Secure wrapper frames which should provide confidentiality, integrity, freshness and authenticity.

KNX IP Secure predominantly uses symmetric cryptography mechanisms. More precisely, it uses the AES-128 as basic block cipher for all modes of operation required in the specification. There are two types of communication in KNX IP Secure, namely unicast communication and multicast communication. In multicast communication, the traffic between members of one group should be secured. Thereby, it exclusively relies on symmetric cryptography schemes. Similar to KNX Data Security, CCM is used as mode of operation. There is also a special case in group communication. This communication type uses a pre-shared secret called Group Key. This key is unique for every IP multicast group. The same group key can be found on every device that is in the same group. A device can only be in one IP multicast group at a time. Unicast traffic is mainly used for configuration purposes, thus securing the communication between a management device and an interconnection device. To achieve perfect forward secrecy, KNX IP Secure uses Elliptic Curve Diffie Hellman (ECDH) key exchange algorithm over NIST curve K-283. Although the draft standard has been proposed recently, KNX IP Secure is still analyzed regarding its security properties (cf. [JKK14].

4 System Architecture

This section presents a KNX extension applicable to environments with increased availability demands for TP based KNX networks. The extension is designed in a transparent way utilizing a ”plug & play” functionality to build a secured KNX network [Gl15]. A standard KNX device outside this secured network should be able to send and receive messages via the secured network without any prerequisites. Every device with one connection to an unsecured KNX TP network and two distinct TP connections to a secured KNX network will act as a KNX security router. Thus, the presence of at least two of these KNX security routers connected to each other by two secured TP lines will constitute a secured KNX area spanning between installations with increased security demands as shown in Figure 2. The secured network is reserved for security routers only, i.e. no standard KNX devices are allowed here.

The basic tasks of the KNX security routers consist of

1. establishing keys with their communication partners within the secured KNX network,
2. providing redundant communication lines, achieving improved availability by encrypting and authenticating all messages which are received on the unsecured line, and delivering them to the proper KNX security router which acts as border device for the given GA, and
3. checking all messages which are received on the secured lines for integrity and authenticity, removing duplicates, unwrapping and delivering them to the unsecured line.

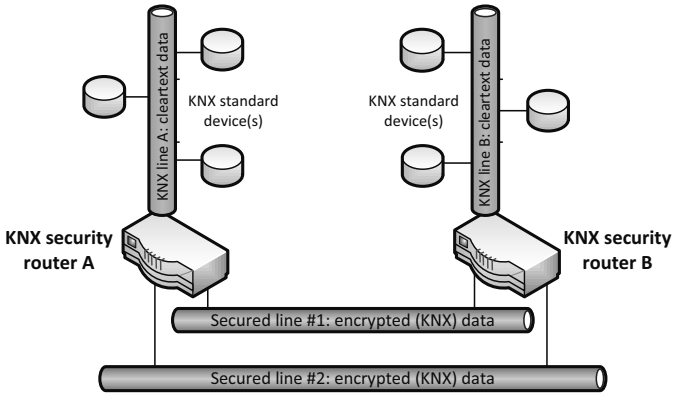


Fig. 2: Unsecured and secured KNX lines

In order to keep the communication overhead as small as possible, an efficient concept for translating KNX addresses to secure KNX addresses (and vice versa) is defined. When a new KNX security router is added to the network, it generates a list of group addresses it acts as border router for. GA out of this list can be addressed later on. This allows to dynamically deploy new KNX security routers with connected unsecured devices achieving a compromise between flexibility and complexity. Sending data to a GA follows the triad of discovery request, discovery response and data transfer, illustrated in Figure 3. Broadcast messages are depicted as solid end of the arrow, while others denote unicast messages.

To enable multiple devices to announce responsibility for a GA, a KNX security router in charge of forwarding data, must accept discovery responses following its own request message for a short time window. The discovery messages generated by KNX security routers have to be encrypted. Although these datagrams do not contain KNX payload per se, they allow an adversary to learn the topology of the network. For example, if an attacker detects that a particular KNX security router is responsible for only one GA and further gets knowledge that this GA is responsible for switching a light (i.e., by visual observation), the attacker afterwards may be able to derive a personal profile just by detecting packets for this GA, although the payload of the datagrams to the responsible KNX security router are encrypted (see below).

Discovery requests are broadcast messages readable by all KNX security routers. To limit the protocol overhead, a global network key is used. For providing authenticity, all datagrams passing the secured KNX network must contain a MAC to prevent modification of them. Defense against replay attacks is achieved by counters. The counters must be strictly monotonically increasing and must not overflow. They can be compared to an initialization vector that prevents the mapping of same cleartext messages to same ciphertext messages under the same encryption key.

Two different types of counters are used: one global counter Ctr_{global} and individual counters Ctr_{ind}. The first one is used for avoiding replay attacks against discovery messages. Ctr_{global} is a 4 byte integer, allowing $2^{32} \approx 4,3$ billion discovery request or response mes-

sages to be sent before overflowing. For synchronizing the global counter Ctr_{global}, a specific service is defined, allowing a newly powered-up device that wants to join the network to synchronize with the rest of the network. Ctr_{ind} is used for the control data transfer. Beside avoiding replay attacks, this counter is necessary to detect and delete duplicates caused by the redundant network. Firstly, it is used as outgoing counter value Ctr_{out} when sending control data from a specific IA. Secondly, every security router maintains a list of incoming individual counter values, Ctr_{in}, also referenced by the unique IAs, thus allowing to discard the duplicates.

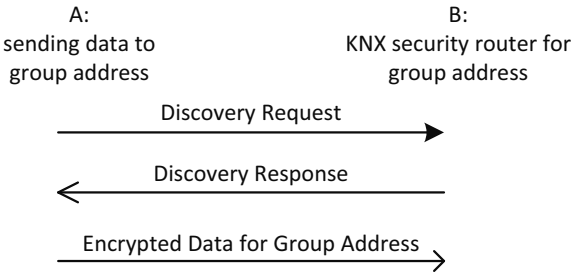


Fig. 3: Discovery and data communication phase

A feasible way to unambiguously identify duplicates is by referencing both Ctr_{out} and Ctr_{in} by the IA of the original cleartext message. This solution works despite potential network faults on one or both secured lines, provided that each KNX device is identified by a unique IA.

While it would be possible to use a centralized concept for the key management, no trusted on-line party is typically available in building automation networks. A centralized approach would need fall-back key servers which inherit the task of generating and distributing keys and parameters in case of a master key server failure. Otherwise, the network would suffer from a single point of failure in case no fall-back mechanism is applied – an assumption that would clearly disqualify the design as highly available. In contrast, the following lean key management is proposed:

- Known by all KNX security routers, a long-term key is defined. This pre-shared key k_{psk} must be copied to every device at setup time and is used for symmetric encryption: (1) it authenticates synchronization messages; (2) it encrypts discovery requests and decrypts discovery responses; (3) it authenticates discovery messages.
- Another pre-shared key is used to authenticate the Diffie-Hellman (DH) parameters, as defined next.
- Asymmetric keys are used for end-to-end encryption of the actual data packets between two KNX security routers. ECDH serves as key negotiation algorithm. To protect against man-in-the-middle attacks, authenticity of the DH parameters must be assured.

A powered-up KNX security router must at first obtain the global counter Ctr_{global} by sending out a synchronization request. This counter is used to avoid deterministic encryp-

tion of discovery messages. To authenticate the joining device and to avoid replay attacks a MAC in combination with a time stamp is used, which means that all KNX security routers must have their system clocks synchronized. To loosen this restriction, a time-window of deviation in the order of seconds is allowed.

If a KNX security router receives a cleartext message, it will at first check the counter value Ctr_{out} for the IA and increment it. After that, the discovery phase takes place by sending request messages. These discovery requests are answered by KNX security routers with discovery responses. Discovery messages must contain the strictly monotonically increasing Ctr_{global}, the referenced GA and will also transport the chosen DH parameters used for later encryption of the actual KNX data frame. Authenticity is assured by a MAC. If at least one valid reply is received (correct MAC and incremented Ctr_{global}), the origin cleartext packet is duplicated, encrypted, put into a unicast data frame together with Ctr_{out} and sent on both lines. The KNX security router in charge of delivering the frame to its KNX segment must maintain a counter for incoming packets (Ctr_{in}) which will be updated by the received counter value. An update occurs as soon as the first frame is received if Ctr_{in} is higher than the stored counter. Subsequently, the delivery of duplicates can be detected because the received counter value will equal the saved counter value. Depending on the availability of the redundant channels, three cases can be distinguished:

- If both secured lines are available, one data frame will be handled first by the receiving KNX security router and the counter will be saved as actual Ctr_{in} for the IA of the inner frame. When handling the second frame, the received counter will equal to the saved counter, and thus the frame will be discarded.
- If only one secured line is available, no duplicate will arrive, but the receiving KNX security router(s) will nevertheless update the received counter value for the IA.
- If both lines are unavailable, the responsible KNX security routers cannot update the corresponding value for Ctr_{in}. Nevertheless, the sending side will further update the outgoing counter Ctr_{out} for the given IA. As soon as the responsible KNX security routers are reachable again, new data frames will contain a higher counter than the one saved on the receiving side, allowing data transfer to the GA again.

5 Discussion

The aim of this section it to show that the communication network is able to resist against maliciously introduced attacks as well as unintended faults happening randomly which results in a KNX network with improved availability. In the case of DoS attacks, the proposed solution can withstand such an attack against one of its two secured communication lines because of the applied redundancy. It is assumed that an attacker only has polynomially bounded processing power and is able to passively read frames from and inject arbitrary frames into both secured communication lines. In contrast, it is assumed that the cleartext KNX lines are out of reach of an attacker. Also, a KNX security router's hardware is expected to be physically secured, in particular the memory holding the long-term encryption and authentication keys.

5.1 Communication overhead

Neglecting the overhead for synchronization messages, one discovery request has to be sent for every cleartext KNX message answered by one response message for every KNX security router handling the GA in question. The exact size of these messages depends on the DH parameters. Additionally, the whole cleartext message (header and payload) will be encrypted and wrapped into another frame together with a MAC, Ctr_{ind} and the header information of the outer frame. This message will be encrypted and sent independently to all KNX security routers that announced responsibility for the GA.

For a large number of KNX security routers responsible for a given GA, the proposed approach introduces a significant network overhead. This overhead could be reduced by using a GA cache. However such a cache was not considered in this proposal, yet.

5.2 Synchronization phase

A KNX security router joining the network must get knowledge of the actual value of Ctr_{global}. This is achieved by sending a broadcast message on every secured line serving as synchronization request. The frame contains the device's local time in seconds. Every device receiving such a request checks the integrity of the message first by recalculating the MAC. Afterwards, freshness is checked by comparing the supplied time with its local time. If the timing information equals the device's own local time, the device returns a synchronization response frame, containing its local time and the actual counter value.

- Passive attacks: Packets in the synchronization phase are not encrypted, allowing a passive adversary to learn the value of the global counter value Ctr_{global}. Nevertheless, this counter is only used to avoid deterministic encryption and is of no use for the attacker.
- Active attacks: An active attacker can inject new synchronization request and response messages, but will fail to produce a correct MAC for the actual time stamp with probability $1 - \frac{1}{2^{32}}$ because the MAC equals a random 32 bit number for the given header and payload. Such a MAC forgery will be detected by all active KNX security routers, and the corresponding frame will be discarded. Opening a window for tolerating clock deviations allows an attacker to resend captured synchronization messages within that time window. In case of a replayed synchronization request message, the attacker can trigger a new synchronization response message by a legitimate KNX security router. The response message will return the actual counter value to the original source address of the replayed message. The corresponding device however has already finished the synchronization phase and will just drop the message. When replaying a synchronization response message within the valid time window, there are two possibilities. If a joining device is waiting for a response message, the replayed message will be handled as legitimate response, and the newly joined device concludes the synchronization phase. On the other hand, if no device is waiting for a synchronization response, the message will simply be dropped.

5.3 Discovery phase

Whenever a KNX router receives a message on its unsecured line, two distinct discovery broadcast messages are sent, one for each secured line. Every device in the network first checks the authenticity of the received frame by recalculating the MAC. The requested GA is obtained by decrypting the corresponding field. Every KNX router acting as a border router for the GA prepares a response frame. During this phase, also ECDH parameter are exchanged. Integrity of the discovery messages is achieved by a MAC.

- Passive attacks: In this phase, a passive attacker is able to learn the global counter value Ctr_{global} as well as the exchanged ECDH parameters. For the first case, the same arguments as given for the synchronization phase hold. For the second case, it can be argued as follows: to derive the key used by two parties in the subsequent data transmission from the DH parameters the attacker would need to solve the Elliptic Curve Discrete Logarithm Problem (ECDLP) [HMV04] – an infeasible task if proper ECDH parameters are chosen. Additionally, the attacker can learn the encrypted value of the requested GA and, thus, discovers which KNX security router(s) act as border routers. However, it is impossible for the attacker to derive the underlying cleartext GA because of the AES encryption, assuming that the attacker does not know the long-term encryption key. Nevertheless, the attacker is able to detect the basic topology of a network.
- Active attacks: For newly generated injected discovery request messages, the attacker must at first generate the encryption for the expected GA. Afterwards, the attacker must additionally guess the correct MAC. Trials to forge the correct MAC will be detected by all receiving devices. Similar arguments hold for discovery response messages. Replaying discovery messages is considered as uncritical because of the freshness property provided by Ctr_{global}. Such repeated frames will be detected by the KNX security router because of the outdated value of Ctr_{global}, which will just drop the replayed frame. Attacking alternating secured communication lines will also fail. For example, the attacker could at first shortcut one secured communication line such that the discovery message will not reach its recipient(s). Receiving the discovery message on the other line and injecting the frame to the previously blocked line would result in a fresh counter value. Alternating the source and/or destination addresses will invalidate the MAC, forcing the attacker again to forge the MAC, an infeasible task as already stated.

5.4 Data transmission phase

During the data transmission phase, a KNX security router in charge of forwarding a KNX message over the secured lines can also derive pairwise shared secrets with all responsible KNX security routers based on ECDH. From this shared secret, a key is derived which is used to encrypt the control data and inserted into the frame after the counter value.

- Passive attacks: An eavesdropping attacker will be able to learn source and destination addresses of the KNX security routers exchanging the frame, as well as the length of the original frame and the individual counter Ctr_{ind}. Again, the meta information can only be used to generate communication profiles, a fact considered inevitable. The individual counter is used as freshness property and to detect the duplicate frames on the receiving side. Therefore, an attacker does not benefit from knowing this counter value. Decrypting the contained inner frame is considered impossible based on the following facts: (1) encryption is based on AES-256, therefore trying all possible keys is infeasible. (2) The attacker is unable to get knowledge of the key because of the key agreement protocol used in the discovery phase.
- Active attacks: An attacker, trying to inject a new data frame, must succeed in forging the correct MAC. A MAC mismatch will be detected by the receiving KNX security router. A replayed message will be correctly verified and decrypted by the receiving device, but because of the outdated counter value Ctr_{ind}, the message will be discarded.

6 Conclusion and Outlook

In order to be able to deploy a communication system in more demanding environments, it is necessary to achieve informational security combined with mechanisms for increasing the availability of a system. KNX is a well-established control network technology tailored to the special needs of buildings. Originally, it was not designed for the use in critical environments. Increasing the security and especially the availability of KNX would allow its deployment also for critical applications. For KNX, extensions for securing a network against malicious attacks exist, but these extensions are not able to handle faults concerning the communication medium as well as DoS attacks. This paper work proposes a solution unifying all three columns of information security, namely availability, confidentiality and integrity, thus protecting against active and passive adversaries including even transient hardware faults. The proposal is even able to resist restricted DoS attacks. This is achieved by using KNX security routers which are connected to each other in a redundant way. A standard KNX line is connected through a KNX security router which copies the received KNX frames into two properly secured frames and sends them over both secured communication lines. The receiving KNX security router will check the incoming frames for modification, discard one of the two copies and forward the remaining one to the destined KNX line.

The proposed solution can withstand malicious attacks as well as transient hardware faults on one of the secured lines. It allows to connect standard KNX devices which are spatially divided in a secure manner, bridging over areas where malicious behavior cannot be ruled out. The approach was inspired by security mechanisms defined by the recent KNX security extensions. However, since network stacks covering these features are not available it was decided to rest the approach currently on its own security mechanisms. The feasibility of the solution was tested by a proof of concept. For the KNX security routers, RaspberryPis in combination with KNX-USB-dongles were used. To interface with KNX,

the client library of eibd[2] was used providing functions for sending and receiving KNX frames. The needed cryptographic functions were built based on OpenSSL[3].

An enhancement for future developments would be to integrate a caching mechanism for the mapping of GA to KNX security routers and encryption keys. A KNX security router receiving data from a KNX device would send a discovery message once and cache the address(es) of the responsible KNX security routers(s), together with the corresponding encryption key(s). Subsequent data transfer can be executed without the need of the discovery phase reducing the bus load. Finally, the presented approach could also be applied to other building automation networks such as LonWorks with slight modifications.

References

[Av04] Avizienis, A.; Laprie, J.-C.; Randell, B.; Landwehr, C.: Basic concepts and taxonomy of dependable and secure computing. Dependable and Secure Computing, IEEE Transactions on, 1(1):11–33, Jan 2004.

[CCM10] Cavalieri, S.; Cutuli, G.; Malgeri, M.: A study on security mechanisms in KNX-based home/building automation networks. In: Emerging Technologies and Factory Automation (ETFA), 2010 IEEE Conference on. pp. 1–4, 9 2010.

[Gl15] Glanzer, Harald: Highly available KNX networks. Master's thesis, TU Wien, 2015.

[GPK10] Granzer, W.; Praus, F.; Kastner, W.: Security in Building Automation Systems. Industrial Electronics, IEEE Transactions on, 57(11):3622–3630, Nov 2010.

[Gr10] Granzer, Wolfgang: Secure Communication in Home and Building Automation Systems. PhD thesis, TU Wien, 2010.

[HMV04] Hankerson, D; Menezes, A.; Vanstone, S.: Guide to Elliptic Curve Cryptopgraphy. Springer, New York, 2004.

[JKK14] Judmayer, A.; Krammer, L.; Kastner, W.: On the security of security extensions for IP-based KNX networks. In: Factory Communication Systems (WFCS), 2014 10th IEEE Workshop on. pp. 1–10, May 2014.

[Ka05] Kastner, W.; Neugschwandtner, G.; Soucek, S.; Newmann, H.M.: Communication Systems for Building Automation and Control. Proceedings of the IEEE, 93(6):1178–1203, 6 2005.

[Kr13] Krammer, Lukas; Bruyne, Steven De; Kastner, Wolfgang; Granzer, Wolfgang: Security Erweiterung für den KNX Standard. In: Tagungsband – innosecure 2013 – Kongress mit Ausstellung für Innovationen in den Sicherheitstechnologien Velbert Heiligenhaus. pp. 31–39, 9 2013. german.

[KS05] Koeune, Franois; Standaert, Franois-Xavier: A Tutorial on Physical Security and Side-Channel Attacks. In (Aldini, Alessandro; Gorrieri, Roberto; Martinelli, Fabio, eds): Foundations of Security Analysis and Design III, volume 3655 of Lecture Notes in Computer Science, pp. 78–108. Springer Berlin Heidelberg, 2005.

[Le09] Lechner, D.; Granzer, W.; Praus, F.; Kastner, W.: Securing IP Backbones in Building Automation Networks. In: Proc. 7th IEEE International Conference on Industrial Informatics (INDIN '09). pp. 410–415, 6 2009.

[2] https://www.auto.tuwien.ac.at/ mkoegler/index.php/eibd

[3] https://www.openssl.org/

M. Meier, D. Reinhardt, S. Wendzel (Hrsg.): Sicherheit 2016,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2016 253

Detecting anomalies in BACnet network data

Jernej Tonejc[1], Jaspreet Kaur[2], Alexandra Kobekova[3]

Abstract: Over the last few years, the volume of data in the Building Automation System (BAS) networks has increased exponentially. Nowadays, it is possible to obtain several kinds of data from building networks such as data based on individual service type, specific building location and even specific time of the day. As a consequence, large volumes of data with more variables have to be considered when performing the data analysis. This means that there is a need to identify the most important variables for analysis. In this paper, we introduce a framework which allows the characterization of BACnet network traffic data by means of machine learning techniques. This framework is based on unsupervised machine learning methods, specifically, Principal Components Analysis and Clustering. Such methods are used because of the large volume of data that needs to be taken into consideration, preventing the manual labeling of the data which is required for supervised learning methods. We show the efficiency and effectiveness of the framework in detecting anomalies by performing experiments on different BACnet network traffic data, captured by Wireshark, together with synthetically generated data.

Keywords: BACnet, data analysis, machine learning, flow mapping, unsupervised learning.

1 Introduction

Building Automation System (BAS) performs the monitoring and controlling of all the electrical and mechanical equipment of a building, from lighting to HVAC to security systems. The structure of BAS in a building is composed of an array of mechanical and electrical devices. All the devices are connected to a central control station, usually a computer, where BAS administrators can get an insight into the whole network. Different kinds of data are sent across the BAS network, including the values when a piece of equipment is turned on and off, change in value when some movement is detected in particular areas of a building, change of temperature within the building spaces, change of temperature of air or water within the mechanical systems, valve positions, etc.

Due to the continuous growth in BAS networks, huge amount of data is transferred through such networks every day. Thus, it is becoming an issue for the BAS administrators to perform efficient data analysis on such a big amount of data. It is hard to identify correlations and detect anomalies on such data traces and sometimes most of the anomalies go undetected by the BAS administrators.

To this end, new efficient techniques capable of dealing with huge BAS network traffic data need to be introduced. This work presents a framework which performs BAS traffic

[1] Fraunhofer FKIE, Cyber Security, Friedrich-Ebert-Allee 144, 53113 Bonn, jernej.tonejc@fkie.fraunhofer.de
[2] Fraunhofer FKIE, Cyber Security, jaspreet.kaur@fkie.fraunhofer.de
[3] Fraunhofer FKIE, Cyber Security, alexandra.kobekova@fkie.fraunhofer.de

analysis by means of machine learning techniques. The framework is graphically depicted in Fig. 1.

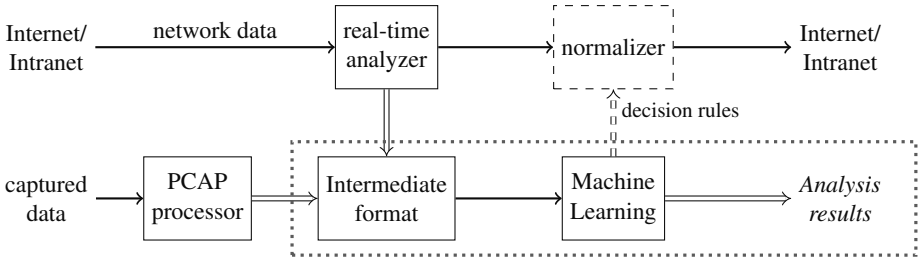


Fig. 1: BAS network analysis framework. The main components of this framework are the *real-time analyzer* and the *normalizer*. Both real-time network data and the captured data are converted into an intermediate form (described in Sect. 3), which is then used to generate decision rules for normalizer with the help of the machine learning methods (described in Sect. 4). This paper focuses on the components within the dotted rectangle.

The main goal of this framework is to characterize the traffic data and detect anomalies. Analysis is done on the BAS traffic captured by `Wireshark`, to differentiate between the relevant traffic and the anomalies. Unsupervised machine learning techniques are used because they are quite efficient in performing analysis on the large volume of network data (and continuously expanding data due to addition of new devices in the network). We consider the following methods: Principal Components Analysis (PCA), and Clustering. These methods have proven to be quite useful in detecting anomalies and discovering patterns while analyzing the network data from other network protocols [DD11].

The network traffic analysis is done on BACnet traffic data. BACnet protocol is one of the data communication protocols used in BAS [ISO12] and has been implemented in products by more than 800 vendors worldwide. The framework supports analyzing prerecorded data in `pcap` format, as well as live capture and on-the-fly analysis using a dedicated analyzer that outputs the live data in the intermediate format.

The rest of the paper is organized as follows. The related work is discussed in Sect. 2. Section 3 explains the structure of BACnet/IP and the machine learning methods used in this study. Section 4 describes the implementation details of machine learning methods along with the flow classifiers. The obtained results are discussed in Sect. 5. The conclusions and future work are presented in Sect. 6.

2 Related Work

Enormous amount of work has been devoted to the application of machine learning and data mining techniques to solve the problem of network traffic analysis. In [NA08], authors present a survey on the application of machine learning techniques for internet traffic classification. Several surveys of anomaly detection methods exist, for example, [C$^+$09]. Researchers have experimented with both supervised learning and unsupervised learning methods because network traffic analysis is rather a broad domain. Some examples of

supervised learning methods to classify network traffic data include Naive Bayesian classifiers [MZ05], and Bayesian neural networks [A+07]. These methods are capable of performing the analysis on a very large set of data but require previous knowledge of the application domain.

Instead, we use unsupervised machine learning methods in our work which can easily identify new anomalies. Unlike supervised learning methods, these techniques do not depend on labelled training data. These approaches are quite simple and effective in supporting the network traffic data analysis. Our analysis mechanism for BAS network traffic data describes the network behavior in a way that enables an extraction of a set of rules. The rules support the administrator to efficiently identify relevant and irrelevant communication among the BAS traffic data. The unsupervised learning methods also have some drawbacks compared to the supervised methods, for example, they generally produce more false positives, and can limit the performance of some specific tasks because of the use of general notions. Some examples of unsupervised learning methods for network traffic data analysis include Clustering algorithms [E+06], Random Forests [DD11], One-Class SVM [DD11] etc.

In another work [M+14], the authors present a comparison of different machine learning methods like Conditional Restricted Boltzmann Machine (CRBM), Artificial Neural Networks (ANNs), and Hidden Markov Models (HMMs) for estimating energy consumption in the buildings. They showed the evaluation of performance of CRBM in the context of building automation systems and proved that CRBM outperforms ANNs and HMMs.

In [P+14], authors propose an anomaly-based intrusion detection system for BACnet/IP networks using Repeated Incremental Pruning to Produce Error Reduction (RIPPER) as a learning method. The features used for learning are similar to the features we use in our methods, however, the algorithm is a supervised learning algorithm and requires the input data to be tagged, and as such, it is less capable of detecting new or unknown types of anomalies.

3 Structure of BACnet/IP data

This section briefly describes the structure of the BACnet/IP network data and introduces the machine learning methods that we use to analyze the data.

3.1 Data selection and processing

In the case of BACnet/IP data over Ethernet, the data is encapsulated in User Datagram Protocol (UDP) layer. Each packet contains communication and control values. An important feature of BACnet networks is the requirement that there exists a unique message path between any two nodes on an interconnected BACnet network [ISO12]. One can therefore speak of flows between BACnet devices, and each flow is defined by the BACnet addresses of the communicating devices. As shown in Fig. 1, the data can either come

from real-time capture or from a prerecorded data. The prerecorded data is captured using `Wireshark` and stored in a raw `pcap` format. In order to convert the raw data into the intermediate format, a fair amount of preprocessing is needed. In case of real-time capture and processing, a dedicated traffic analyzer outputs the data directly in the intermediate format. Figure 2 shows the structure of a typical BACnet/IP packet.

Fig. 2: BACnet/IP data is contained within the UDP datagram as a BACnet Virtual Link Layer (BVLL), which contains the Network layer Protocol Data Unit (NPDU) and the Application layer Protocol Data Unit (APDU) with data.

For a detailed explanation of the structure of NPDU and APDU, we refer the reader to [K+15]. For our analysis we chose the same fields as were suggested in [T+15]. For the sake of completeness, we list the fields in Fig. 3, grouped by the layer in which they appear. The fields that are present in every valid BACnet/IP packet are shown in bold.

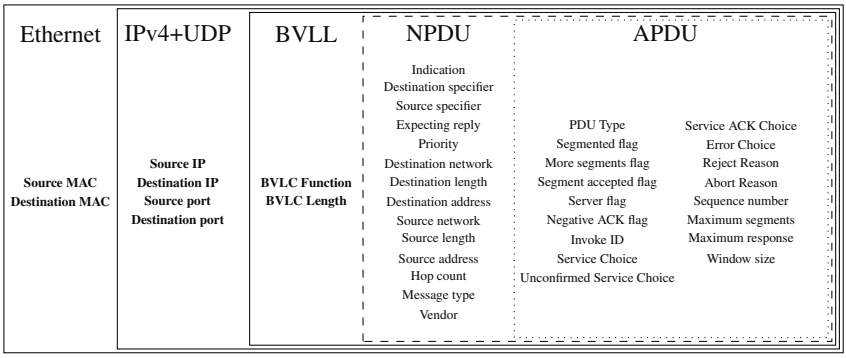


Fig. 3: The fields as features, organized by the layers in which they are present. In addition to the above fields, a timestamp is recorded for each packet (both in recorded `pcap` data as well as during the real-time analysis), bringing the total number of features per packet to 40.

BACnet packets do not have a fixed structure, as the presence of many fields depends on the values of various control bits and fields. For example, the APDU fields Service Choice, Unconfirmed Service Choice, Error Choice, Reject Reason and Abort Reason are never present at the same time, as each exists only for a specific value of the PDU Type field. We simply use the value zero for the missing fields.

The selected fields contain all the essential features of each BACnet/IP packet. Before we can use the network data in machine learning methods, we need to process it. The purpose of processing the network data is twofold:

1. to prepare the data for subsequent machine learning analysis, and
2. to aggregate and export the data in a form suitable for visualizing the flows and creating the model for the flow-based anomaly detection.

All of the fields in BACnet/IP data have a natural representation as unsigned integers. We perform only the following processing. We split the MAC addresses into the Organizationally Unique Identifier part and the vendor-specific identifier, each using three bytes of the MAC address, and encoded as 32-bit integers. The IP addresses are encoded as four 8-bit integers to reflect the natural structure of IP-based networks. The BACnet source and destination addresses could be up to 7 bytes long, so we encode them using 64-bit integers. In case of a malformed packet with address longer than 8 bytes we truncate the value to 64 bits. There is no loss of information for valid packets and the anomalies in addresses can still be detected as the value of the address length field falls outside the valid range. These preprocessing steps convert the 40 features into 48 numerical columns.

Due to completely different scales of data in various fields, e.g., the values for valid BACnet addresses could vary from 0 to 72057594037927935, for Hop count from 0 to 255 and for Indication from 0 to 1, we need to normalize the values, as some of the machine learning methods rely on Euclidean distance between the data points. In particular, PCA and distance-based clustering methods are affected by disproportionate scale for different features.

3.2 Machine learning methods

In order to detect new and unknown traffic anomalies, we need to use unsupervised machine learning methods, as the sheer volume of data prevents any manual tagging of individual packets. There are various machine learning methods that are well-suited for anomaly detection in the network traffic [C$^+$09, DD11]. In this paper, we focused on the following:

1. *Clustering methods*. We focus on the distance-based clustering methods, in particular, *k*-means and expectation maximization (EM). In *k*-means, the data is partitioned into *k* clusters, such that a point in a given cluster is closer to that cluster's centroid than to any other cluster's centroid. For EM, one first needs a probabilistic model, for which the parameters need to be estimated. The parameters are then determined in such a way as to maximize the expectation of the log likelihood.
2. *Principal Component Analysis*. Each packet is treated as a point in a 48-dimensional space. The objective is to project the data on a lower dimensional space, retaining as much variance as possible.

The principal component analysis was used as a preprocessing step before the clustering. The clustering methods were used as a classifier, where the small enough clusters were marked as anomalous.

3.3 Network flows

The communication in BACnet can either be a direct communication between two devices or a broadcast message (local, remote or global). The broadcast messages can be

viewed as communication with a special device and are simply treated the same way as other communication in our analysis. The flows can be defined in terms of the source and destination MAC addresses, source and destination IP addresses or source and destination BACnet addresses. Different underlying structures of the BACnet network are revealed when using different flow definitions. For each such flow, a multidimensional statistical model is constructed to characterize the types of messages and their probabilities of occurrence, as well as to determine the expected packet sizes. Due to the mostly static nature of the building automation networks, such flow maps enable efficient anomaly detection. For the flow-based anomaly detection to work we need to make sure we can map the network when no anomalies are present, to obtain the base state of the network. The anomalies can later be detected using analytic methods, where we directly compare the current flow map with the original one, as well as the statistical properties of the individual flows, such as the message type probability distribution, and the packet frequency based on the flow and message type.

4 Implementation Details

In this section, we describe the details of the implementation, together with the data sources that were used to train and test our methods.

4.1 Data sources

As mentioned in Sect. 3, the `pcap` data needs to be processed to make it accessible to the machine learning methods. The preprocessing was implemented in C for efficiency reasons, following the BACnet standard [ISO12]. The result of the preprocessing is the intermediate format, which is simply a CSV-file with 48 columns. The same format is used when analyzing the packets in real time. For certain machine learning methods, namely *k*-means, EM and PCA, we also scale the data in each columns so as to make the values in that column lie between 0 and 1.

We use two types of sources for training and evaluating our methods. The first source is our BAS lab, which contains several BACnet devices. The traffic is recorded using Wireshark on a PC which is connected to the mirror port of a local switch. We recorded two days worth of traffic. On the first day, only the normal behavior of the lab was present, i.e., no deliberate anomalies were introduced. On the second day, we performed several scans of the BACnet network using a free demo version of a tool called BACeye. The two pcap files contain about 1.2 million BACnet packets each. The preprocessing step took about 6 seconds in total. The amount and variety of data is limited due to the small size of our BAS lab. For comparison, the number of packets that we observed in a large university network is about 16 million per day, or about ten times more than in our lab setup, so the lab setup is a relatively good approximation of an actual building automation network.

The second source of data is a set of artificially generated intermediate format files. As a basis we take the traffic recordings from our lab for a different day, then add various levels

of synthetically generated anomalous traffic. The proportion of the anomalous traffic is 0.1%, 0.5%, 1% and 2%, with anomalous records either bunched up or roughly uniformly dispersed throughout the files (this affects the timing information of those packets). We analyzed the attacks in [K+15] and based our anomalies on the following possible attacks:

- Covert channels using different message types to convey information (CCM)
- Covert channels using reserved bits (CCR)
- Using APDU service choices for writing properties or modifying objects (WP)
- Using incorrect bit combinations and strange values for BACnet address lengths (IB)

We generated 100,000 records for each combination, for a total of 33 (=$1+2\times4\times4$) files, including the file without modifications.

4.2 Implementation of machine learning methods and validation

The analysis of the network data was done after the preprocessing step. For the machine learning methods, we extracted all 40 listed features from each BACnet/IP packet, performed the preprocessing steps as indicated in Sect. 3 and obtained a 48-dimensional vector. We then used Weka [H+09] to perform the selected machine learning algorithms. We performed the algorithms on large enough traffic recordings to capture the typical network behavior. Later, we check for each additional packet how well it fits into the model that was learned. This allows us to detect anomalies in the traffic as they do not fit the parameters that were extracted from the sample data. For the learning to be successful, the training data really needs to be representative, otherwise too many false positives are detected.

The data was split into a training set and a testing set. For the lab data we split it into 100,000-line chunks, then trained the algorithms on a randomly selected 66% of the data in one chunk, performed validation on the remaining 34% of the same chunk and then tested the remaining chunks. For the synthetically generated data, we performed the same 66%–34% training/validation split on the file with 1% of anomalous data, then tested the files with different proportions of anomalous data. In addition, we combined the 1% files with all the different types of anomalies and trained the methods on that (with the same training/validation split), then tested all the individual files to evaluate the ability of the so-trained methods to detect various anomalies.

We performed the following machine learning algorithms:

- k-means with $k = 5$ (km5) and $k = 10$ (km10) clusters
- EM clustering with $k = 5$ (EM5) and $k = 10$ (EM10) clusters
- PCA, followed by k-means with $k = 5$ (Pkm5) and $k = 10$ (Pkm10) clusters
- PCA, followed by EM clustering with $k = 5$ (PEM5) and $k = 10$ (PEM10) clusters

As stated in Sect. 3, we used the clustering methods as classifier, where we marked every cluster smaller than 2% of the total data as anomalous. This was based on the assumption that at most 2% of the data is anomalous. This can potentially lead to many false positive results if the number of clusters is too large, since the non-anomalous clusters also get smaller as the total number of clusters increases.

To evaluate the quality of the chosen machine learning methods, we used the standard quality measures, such as sensitivity, specificity, precision, accuracy and the F_1-score, derived from the *confusion matrix*

	Packet is anomalous	Packet is normal
Packet predicted anomalous	T_P	F_P
Packet predicted normal	F_N	T_N

where T_P denotes the true positives, T_N true negatives, F_P false positives, and F_N false negatives. The sum of all four quantities equals the number of the analyzed packets.

The F_1 score is the harmonic mean of the sensitivity and precision and given the expected nature of the anomalous data, we expect that the accuracy does not say much about the quality of the machine learning methods. Therefore we use the specificity (SPC) and the F_1 score, together with the Matthews correlation coefficient, which is considered one of the best measures for two-class classification [Pow11]. The three measures can can be computed directly from the above confusion matrix as

$$\mathrm{SPC} = \frac{T_N}{F_P + T_N},\ F_1 = \frac{2T_P}{2T_P + F_P + F_N},\ M_{CC} = \frac{T_P T_N - F_N F_P}{\sqrt{(T_P + F_N)(T_P + F_P)(T_N + F_N)(T_N + F_P)}}.$$

4.3 Analyzing the flow data

As mentioned in Sect. 3, when analyzing the flows, the packets can be grouped based on their source and destination addresses. Additionally, the packets can be split into two sets: packets without APDU (these are network layer messages) and packets with APDU. These two sets give rise to the network-layer flows and application-layer flows. Each device that sent or received a message represents a node in a directed weighted *flow graph*. The nodes can be identified using their MAC addresses, their IP addresses or their BACnet addresses. The directed edges of the graph are the flows. Different choices of the node identifiers and types of flows give rise to several different flow graphs for the same packet recording. Additionally, we can assign weights to the edges based on various features, for example, normalized total number of messages in that particular flow, total number of bytes exchanged, the inverse of the average inter-packet arrival time, etc. The graph data is exported as *Graph Exchange XML Format* (GEXF), as well as Javascript Object Notation (JSON) format.

We can learn the normal state of the BAS network by analyzing the data for which we know there are no anomalies. Later, we can perform anomaly detection by comparing the

current flow graphs with the saved ones. We visualize the flow graphs and the differences using the Javascript framework D3.js. In addition to comparing the differences between the flow graphs, we also run a community detection algorithm on each graph. The algorithm used is from [B+08]. It attempts to maximize the modularity Q of the graph, defined as

$$Q = \frac{1}{M} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{M} \right] \delta(c_i, c_j),$$

where A_{ij} is the weight of the edge between nodes i and j, k_i is the sum of the weights on the outgoing edges at node i, c_i is the community to which node i is assigned, M is the sum of all weights of all edges and δ is the standard Kronecker δ-function. The algorithm starts with each node in its own community and proceeds iteratively by assigning a node to the community that maximizes the increase of the modularity at each step. When assigning the weights to the edges it is important to normalize them to take into account the specifics of BAS, as there is usually more activity during business hours as compared to the night or weekend.

5 Results

In this section, we present the results of the various analyses that were performed on the data.

5.1 Evaluation of machine learning methods

As stated in Sect. 4, we use the standard quality measures for machine learning methods, together with the Matthews correlation coefficient. The results for various methods are presented in Tab. 1. Due to the space constraints, only a subset of the results is shown. The results for 10 clusters were not significantly better in any case and in some cases (specifically, PEM10), the results were actually worse since some small clusters with non-anomalous data were classified as anomalous, increasing the number of false positives. When training with one type of an attack, only the results of testing against the data with 0.1% of anomalies are shown. The results for 0.5%, 1% and 2% are generally better. When training against all the attacks at once, the results for 0.1% and 2% are given.

Since we know what kind of anomalies were introduced into the data, we are able to compute the values of the confusion matrix. We expect that the methods perform similarly when applied to the data for which the anomalies are truly unknown. For all the data sets, between 1 and 3 clusters were marked as anomalous, with 2 clusters (out of 5) in majority of the cases. For k-means, the training step took less than 3 seconds in all cases, for both $k = 5$ and $k = 10$. For EM, the training took about 40 seconds for $k = 5$ and about 80 seconds for $k = 10$, for each dataset. When combined with the PCA, the training for EM took 3, respectively 12 seconds. Testing was in all cases fast. When using PCA, we always took the first three principal directions, covering about 70–80% of the variance. From the table it is clear that by far the best method is k-means with $k = 5$, preceded by PCA. The

Algorithm	Train set	Test set	M_{CC}	Specificity	F_1 score
km5	IB	IB-0.1%	1.0	1.0	1.0
	WP	WP-0.1%	0.9284	0.9998	0.9259
	CCM	CCM-0.1%	0.8908	0.9997	0.885
	CCR	CCR-0.1%	1.0	1.0	1.0
	AA	IB-0.1%	0.9284	0.9998	0.9259
		IB-2%	0.9959	0.9998	0.996
		WP-0.1%	0.9284	0.9998	0.9259
		WP-2%	0.9962	0.9998	0.9963
		CCM-0.1%	0.9284	0.9998	0.9259
		CCM-2%	0.9959	0.9998	0.996
		CCR-0.1%	0.9284	0.9998	0.9259
		CCR-2%	0.9959	0.9998	0.996
	BACeye	BACeye	0.9912	0.9998	0.9912
		no-BACeye	—	0.9998	0.0
EM5	IB	IB-0.1%	0.245	0.9985	0.2357
	WP	WP-0.1%	0.6265	0.9998	0.6215
	CCM	CCM-0.1%	0.8511	0.9996	0.8403
	CCR	CCR-0.1%	0.8511	0.9996	0.8403
	AA	IB-0.1%	0.8157	0.9998	0.8155
		IB-2%	0.8884	0.9998	0.8856
		WP-0.1%	0.8999	0.9998	0.8959
		WP-2%	0.9818	0.9998	0.9821
		CCM-0.1%	0.533	0.9998	0.5212
		CCM-2%	0.7078	0.9998	0.6763
		CCR-0.1%	0.6042	0.9998	0.5977
		CCR-2%	0.7	0.9998	0.6664
	BACeye	BACeye	0.9225	0.9985	0.9202
		no-BACeye	—	0.9986	0.0
Pkm5	IB	IB-0.1%	0.9284	0.9998	0.9259
	WP	WP-0.1%	0.9284	0.9998	0.9259
	CCM	CCM-0.1%	0.9284	0.9998	0.9259
	CCR	CCR-0.1%	0.9284	0.9998	0.9259
	AA	IB-0.1%	1.0	1.0	1.0
		IB-2%	1.0	1.0	1.0
		WP-0.1%	1.0	1.0	1.0
		WP-2%	1.0	1.0	1.0
		CCM-0.1%	1.0	1.0	1.0
		CCM-2%	1.0	1.0	1.0
		CCR-0.1%	1.0	1.0	1.0
		CCR-2%	1.0	1.0	1.0
	BACeye	BACeye	0.9933	0.9999	0.9934
		no-BACeye	—	0.9999	0.0
PEM5	IB	IB-0.1%	0.8511	0.9996	0.8403
	WP	WP-0.1%	0.8908	0.9997	0.885
	CCM	CCM-0.1%	0.8908	0.9997	0.885
	CCR	CCR-0.1%	0.8908	0.9997	0.885
	AA	IB-0.1%	0.8573	0.9996	0.8475
		IB-2%	0.9909	0.9996	0.9911
		WP-0.1%	0.8573	0.9996	0.8475
		WP-2%	0.9912	0.9996	0.9913
		CCM-0.1%	0.8573	0.9996	0.8475
		CCM-2%	0.9914	0.9997	0.9916
		CCR-0.1%	0.8573	0.9996	0.8475
		CCR-2%	0.9917	0.9997	0.9918
	BACeye	BACeye	0.9825	0.9997	0.9825
		no-BACeye	—	0.9996	0.0

Tab. 1: The quality measures for the various machine learning algorithms. The methods and data sets are defined in Sect. 4. AA stands for All Attack types. Since no-BACeye test data contains no true positives or false negatives, the Matthews coefficient cannot be computed in this case.

best results were obtained when the classifier was trained on all the attack types at once. The cluster centers that are obtained in the training step can later be used for real-time analysis of each incoming packet.

5.2 Visualization of the flow data

In addition to detecting anomalies using machine learning methods, we can detect anomalies by constructing directed flow graphs and running the community detection algorithm on the graph, in order to obtain the large-scale structure of the graph. Figure 4 shows the difference between the flow graph of a network without anomalies (left) and the flow graph when the program BACeye is run at some point in time (right). The anomalies are detected and marked automatically by comparing the current flow graph with the original one. We compare the nodes, the edges, and the statistical properties of each edge that appears in both flow graphs.

Fig. 4: Flow graph of a network without anomalies (left) and with anomalies introduced by running the program BACeye. Clearly visible are the new nodes, new flows, and the change in size of the existing nodes.

6 Conclusions and Future Work

We have shown that machine learning methods can be effective in detecting BAS network traffic anomalies. In addition, the communication flows have natural structure that lends itself to an efficient probabilistic description, which simplifies the detection of anomalies. Combined with dynamic visualization techniques, it enables the operators of BAS to detect anomalies early and act accordingly.

We focused on the following unsupervised machine learning methods: *k*-means and expectation maximization, both with or without a preceding principal component analysis. There are other algorithms that could also be used, for example one-class support vector machine, random forests, density-based clustering algorithms like CLIQUE and MAFIA, self-organizing maps using artificial neural networks, as well as a whole class of supervised and semi-supervised methods, for example *k*-nearest neighbor, Kalman filters, and Bayesian networks among others.

The selected features mostly come from the header values of the packets in our analysis, i.e., they primarily reflect the structure of the network and the kinds of communication

and not so much the actual application data. We intend to include more features from the application level in the future, in order to detect anomalies in the *performance* of the hardware controlled by the BAS and not just the communication anomalies. This is important when trying to detect and thwart more sophisticated cyberattacks, where the attackers try to affect the BAS hardware.

Our plan is to test the developed methods on real-life traffic recordings, and to expand our test setup to include more devices with a more diverse set of anomalies. We also plan to implement the decision rule extraction in a way that can be directly used in a real-time traffic normalizer. Both flow-based rules and machine learning-based rules will be extracted.

Acknowledgement. This work was partially supported by the German Federal Ministry of Education and Research (BMBF) through project BARNI, project number 16KIS0148.

References

[A+07] T. Auld et al. Bayesian Neural Networks for Internet Traffic Classification. *IEEE Transactions on Neural Networks*, 18(1):223–239, Jan 2007.

[B+08] V.D. Blondel et al. Fast unfolding of communities in large networks. *J. Stat. Mech.*, 2008(10):P10008, 2008.

[C+09] V. Chandola et al. Anomaly Detection: A Survey. *ACM Comp. Surv.*, 41(3):15:1–58, 2009.

[DD11] S. Dua and X. Du. *Data Mining and Machine Learning in Cybersecurity*. CRC, 2011.

[E+06] J. Erman et al. Traffic Classification Using Clustering Algorithms. In *Proc. 2006 SIGCOMM MineNet '06*, pages 281–286, New York, NY, USA, 2006. ACM.

[H+09] M. Hall et al. The WEKA Data Mining Software. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[ISO12] ISO. Building automation and control systems – Part 5: Data communication protocol. ISO 16484-5:2012, International Organization for Standardization, 2012.

[K+15] J. Kaur et al. Securing BACnet's Pitfalls. In *Proc. 30. IFIP SEC, Hamburg*, volume 455, pages 616–629. Springer, 2015.

[M+14] E. Mocanu et al. Comparison of machine learning methods for estimating energy consumption in buildings. In *PMAPS 2014*, pages 1–6, 2014.

[MZ05] A.W. Moore and D. Zuev. Internet Traffic Classification Using Bayesian Analysis Techniques. In *Proc. 2005 SIGMETRICS '05*, pages 50–60, New York, 2005. ACM.

[NA08] T.T.T. Nguyen and G. Armitage. A Survey of Techniques for Internet Traffic Classification Using Machine Learning. *Commun. Surveys Tuts.*, 10(4):56–76, October 2008.

[P+14] Z. Pan et al. Anomaly based intrusion detection for Building Automation and Control networks. In *Proc. 11th AICCSA*, pages 72–77, 2014.

[Pow11] D.M.W. Powers. Evaluation: From Precision, Recall And F-Measure To ROC, Informedness, Markedness & Correlation. *J. Mach. Learning Tech.*, 2(1):37–63, 2011.

[T+15] J. Tonejc et al. Visualizing BACnet Data to Facilitate Humans in Building-Security Decision-Making. In *Proc. 3rd Int. Conf. HCI-HAS, Los Angeles*, pages 693–704, 2015.

GI-Edition Lecture Notes in Informatics

P-1 Gregor Engels, Andreas Oberweis, Albert Zündorf (Hrsg.): Modellierung 2001.

P-2 Mikhail Godlevsky, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications, ISTA'2001.

P-3 Ana M. Moreno, Reind P. van de Riet (Hrsg.): Applications of Natural Lan-guage to Information Systems, NLDB'2001.

P-4 H. Wörn, J. Mühling, C. Vahl, H.-P. Meinzer (Hrsg.): Rechner- und sensorgestützte Chirurgie; Workshop des SFB 414.

P-5 Andy Schürr (Hg.): OMER – Object-Oriented Modeling of Embedded Real-Time Systems.

P-6 Hans-Jürgen Appelrath, Rolf Beyer, Uwe Marquardt, Heinrich C. Mayr, Claudia Steinberger (Hrsg.): Unternehmen Hochschule, UH'2001.

P-7 Andy Evans, Robert France, Ana Moreira, Bernhard Rumpe (Hrsg.): Practical UML-Based Rigorous Development Methods – Countering or Integrating the extremists, pUML'2001.

P-8 Reinhard Keil-Slawik, Johannes Magenheim (Hrsg.): Informatikunterricht und Medienbildung, INFOS'2001.

P-9 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Innovative Anwendungen in Kommunikationsnetzen, 15. DFN Arbeitstagung.

P-10 Mirjam Minor, Steffen Staab (Hrsg.): 1st German Workshop on Experience Management: Sharing Experiences about the Sharing Experience.

P-11 Michael Weber, Frank Kargl (Hrsg.): Mobile Ad-Hoc Netzwerke, WMAN 2002.

P-12 Martin Glinz, Günther Müller-Luschnat (Hrsg.): Modellierung 2002.

P-13 Jan von Knop, Peter Schirmbacher and Viljan Mahni_ (Hrsg.): The Changing Universities – The Role of Technology.

P-14 Robert Tolksdorf, Rainer Eckstein (Hrsg.): XML-Technologien für das Semantic Web – XSW 2002.

P-15 Hans-Bernd Bludau, Andreas Koop (Hrsg.): Mobile Computing in Medicine.

P-16 J. Felix Hampe, Gerhard Schwabe (Hrsg.): Mobile and Collaborative Business 2002.

P-17 Jan von Knop, Wilhelm Haverkamp (Hrsg.): Zukunft der Netze –Die Verletzbarkeit meistern, 16. DFN Arbeitstagung.

P-18 Elmar J. Sinz, Markus Plaha (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2002.

P-19 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3. Okt. 2002 in Dortmund.

P-20 Sigrid Schubert, Bernd Reusch, Norbert Jesse (Hrsg.): Informatik bewegt – Informatik 2002 – 32. Jahrestagung der Gesellschaft für Informatik e.V. (GI) 30.Sept.-3. Okt. 2002 in Dortmund (Ergänzungsband).

P-21 Jörg Desel, Mathias Weske (Hrsg.): Promise 2002: Prozessorientierte Methoden und Werkzeuge für die Entwicklung von Informationssystemen.

P-22 Sigrid Schubert, Johannes Magenheim, Peter Hubwieser, Torsten Brinda (Hrsg.): Forschungsbeiträge zur "Didaktik der Informatik" – Theorie, Praxis, Evaluation.

P-23 Thorsten Spitta, Jens Borchers, Harry M. Sneed (Hrsg.): Software Management 2002 – Fortschritt durch Beständigkeit

P-24 Rainer Eckstein, Robert Tolksdorf (Hrsg.): XMIDX 2003 – XML-Technologien für Middleware – Middleware für XML-Anwendungen

P-25 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Commerce – Anwendungen und Perspektiven – 3. Workshop Mobile Commerce, Universität Augsburg, 04.02.2003

P-26 Gerhard Weikum, Harald Schöning, Erhard Rahm (Hrsg.): BTW 2003: Datenbanksysteme für Business, Technologie und Web

P-27 Michael Kroll, Hans-Gerd Lipinski, Kay Melzer (Hrsg.): Mobiles Computing in der Medizin

P-28 Ulrich Reimer, Andreas Abecker, Steffen Staab, Gerd Stumme (Hrsg.): WM 2003: Professionelles Wissensmanagement – Er-fahrungen und Visionen

P-29 Antje Düsterhöft, Bernhard Thalheim (Eds.): NLDB'2003: Natural Language Processing and Information Systems

P-30 Mikhail Godlevsky, Stephen Liddle, Heinrich C. Mayr (Eds.): Information Systems Technology and its Applications

P-31 Arslan Brömme, Christoph Busch (Eds.): BIOSIG 2003: Biometrics and Electronic Signatures

P-32 Peter Hubwieser (Hrsg.): Informatische Fachkonzepte im Unterricht – INFOS 2003

P-33 Andreas Geyer-Schulz, Alfred Taudes (Hrsg.): Informationswirtschaft: Ein Sektor mit Zukunft

P-34 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 1)

P-35 Klaus Dittrich, Wolfgang König, Andreas Oberweis, Kai Rannenberg, Wolfgang Wahlster (Hrsg.): Informatik 2003 – Innovative Informatikanwendungen (Band 2)

P-36 Rüdiger Grimm, Hubert B. Keller, Kai Rannenberg (Hrsg.): Informatik 2003 – Mit Sicherheit Informatik

P-37 Arndt Bode, Jörg Desel, Sabine Rathmayer, Martin Wessner (Hrsg.): DeLFI 2003: e-Learning Fachtagung Informatik

P-38 E.J. Sinz, M. Plaha, P. Neckel (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2003

P-39 Jens Nedon, Sandra Frings, Oliver Göbel (Hrsg.): IT-Incident Management & IT-Forensics – IMF 2003

P-40 Michael Rebstock (Hrsg.): Modellierung betrieblicher Informationssysteme – MobIS 2004

P-41 Uwe Brinkschulte, Jürgen Becker, Dietmar Fey, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle, Thomas Runkler (Edts.): ARCS 2004 – Organic and Pervasive Computing

P-42 Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Economy – Transaktionen und Prozesse, Anwendungen und Dienste

P-43 Birgitta König-Ries, Michael Klein, Philipp Obreiter (Hrsg.): Persistance, Scalability, Transactions – Database Mechanisms for Mobile Applications

P-44 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): Security, E-Learning. E-Services

P-45 Bernhard Rumpe, Wofgang Hesse (Hrsg.): Modellierung 2004

P-46 Ulrich Flegel, Michael Meier (Hrsg.): Detection of Intrusions of Malware & Vulnerability Assessment

P-47 Alexander Prosser, Robert Krimmer (Hrsg.): Electronic Voting in Europe – Technology, Law, Politics and Society

P-48 Anatoly Doroshenko, Terry Halpin, Stephen W. Liddle, Heinrich C. Mayr (Hrsg.): Information Systems Technology and its Applications

P-49 G. Schiefer, P. Wagner, M. Morgenstern, U. Rickert (Hrsg.): Integration und Datensicherheit – Anforderungen, Konflikte und Perspektiven

P-50 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 1) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm

P-51 Peter Dadam, Manfred Reichert (Hrsg.): INFORMATIK 2004 – Informatik verbindet (Band 2) Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 20.-24. September 2004 in Ulm

P-52 Gregor Engels, Silke Seehusen (Hrsg.): DELFI 2004 – Tagungsband der 2. e-Learning Fachtagung Informatik

P-53 Robert Giegerich, Jens Stoye (Hrsg.): German Conference on Bioinformatics – GCB 2004

P-54 Jens Borchers, Ralf Kneuper (Hrsg.): Softwaremanagement 2004 – Outsourcing und Integration

P-55 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): E-Science und Grid Adhoc-Netze Medienintegration

P-56 Fernand Feltz, Andreas Oberweis, Benoit Otjacques (Hrsg.): EMISA 2004 – Informationssysteme im E-Business und E-Government

P-57 Klaus Turowski (Hrsg.): Architekturen, Komponenten, Anwendungen

P-58 Sami Beydeda, Volker Gruhn, Johannes Mayer, Ralf Reussner, Franz Schweiggert (Hrsg.): Testing of Component-Based Systems and Software Quality

P-59 J. Felix Hampe, Franz Lehner, Key Pousttchi, Kai Ranneberg, Klaus Turowski (Hrsg.): Mobile Business – Processes, Platforms, Payments

P-60 Steffen Friedrich (Hrsg.): Unterrichtskonzepte für inforrmatische Bildung

P-61 Paul Müller, Reinhard Gotzhein, Jens B. Schmitt (Hrsg.): Kommunikation in verteilten Systemen

P-62 Federrath, Hannes (Hrsg.): „Sicherheit 2005“ – Sicherheit – Schutz und Zuverlässigkeit

P-63 Roland Kaschek, Heinrich C. Mayr, Stephen Liddle (Hrsg.): Information Systems – Technology and ist Applications

P-64 Peter Liggesmeyer, Klaus Pohl, Michael Goedicke (Hrsg.): Software Engineering 2005

P-65 Gottfried Vossen, Frank Leymann, Peter Lockemann, Wolffried Stucky (Hrsg.): Datenbanksysteme in Business, Technologie und Web

P-66 Jörg M. Haake, Ulrike Lucke, Djamshid Tavangarian (Hrsg.): DeLFI 2005: 3. deutsche e-Learning Fachtagung Informatik

P-67 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 1)

P-68 Armin B. Cremers, Rainer Manthey, Peter Martini, Volker Steinhage (Hrsg.): INFORMATIK 2005 – Informatik LIVE (Band 2)

P-69 Robert Hirschfeld, Ryszard Kowalcyk, Andreas Polze, Matthias Weske (Hrsg.): NODe 2005, GSEM 2005

P-70 Klaus Turowski, Johannes-Maria Zaha (Hrsg.): Component-oriented Enterprise Application (COAE 2005)

P-71 Andrew Torda, Stefan Kurz, Matthias Rarey (Hrsg.): German Conference on Bioinformatics 2005

P-72 Klaus P. Jantke, Klaus-Peter Fähnrich, Wolfgang S. Wittig (Hrsg.): Marktplatz Internet: Von e-Learning bis e-Payment

P-73 Jan von Knop, Wilhelm Haverkamp, Eike Jessen (Hrsg.): “Heute schon das Morgen sehen“

P-74 Christopher Wolf, Stefan Lucks, Po-Wah Yau (Hrsg.): WEWoRC 2005 – Western European Workshop on Research in Cryptology

P-75 Jörg Desel, Ulrich Frank (Hrsg.): Enterprise Modelling and Information Systems Architecture

P-76 Thomas Kirste, Birgitta König-Riess, Key Pousttchi, Klaus Turowski (Hrsg.): Mobile Informationssysteme – Potentiale, Hindernisse, Einsatz

P-77 Jana Dittmann (Hrsg.): SICHERHEIT 2006

P-78 K.-O. Wenkel, P. Wagner, M. Morgenstern, K. Luzi, P. Eisermann (Hrsg.): Land- und Ernährungswirtschaft im Wandel

P-79 Bettina Biel, Matthias Book, Volker Gruhn (Hrsg.): Softwareengineering 2006

P-80 Mareike Schoop, Christian Huemer, Michael Rebstock, Martin Bichler (Hrsg.): Service-Oriented Electronic Commerce

P-81 Wolfgang Karl, Jürgen Becker, Karl-Erwin Großpietsch, Christian Hochberger, Erik Maehle (Hrsg.): ARCS´06

P-82 Heinrich C. Mayr, Ruth Breu (Hrsg.): Modellierung 2006

P-83 Daniel Huson, Oliver Kohlbacher, Andrei Lupas, Kay Nieselt and Andreas Zell (eds.): German Conference on Bioinformatics

P-84 Dimitris Karagiannis, Heinrich C. Mayr, (Hrsg.): Information Systems Technology and its Applications

P-85 Witold Abramowicz, Heinrich C. Mayr, (Hrsg.): Business Information Systems

P-86 Robert Krimmer (Ed.): Electronic Voting 2006

P-87 Max Mühlhäuser, Guido Rößling, Ralf Steinmetz (Hrsg.): DELFI 2006: 4. e-Learning Fachtagung Informatik

P-88 Robert Hirschfeld, Andreas Polze, Ryszard Kowalczyk (Hrsg.): NODe 2006, GSEM 2006

P-90 Joachim Schelp, Robert Winter, Ulrich Frank, Bodo Rieger, Klaus Turowski (Hrsg.): Integration, Informationslogistik und Architektur

P-91 Henrik Stormer, Andreas Meier, Michael Schumacher (Eds.): European Conference on eHealth 2006

P-92 Fernand Feltz, Benoît Otjacques, Andreas Oberweis, Nicolas Poussing (Eds.): AIM 2006

P-93 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 1

P-94 Christian Hochberger, Rüdiger Liskowsky (Eds.): INFORMATIK 2006 – Informatik für Menschen, Band 2

P-95 Matthias Weske, Markus Nüttgens (Eds.): EMISA 2005: Methoden, Konzepte und Technologien für die Entwicklung von dienstbasierten Informationssystemen

P-96 Saartje Brockmans, Jürgen Jung, York Sure (Eds.): Meta-Modelling and Ontologies

P-97 Oliver Göbel, Dirk Schadt, Sandra Frings, Hardo Hase, Detlef Günther, Jens Nedon (Eds.): IT-Incident Mangament & IT-Forensics – IMF 2006

P-98 Hans Brandt-Pook, Werner Simonsmeier und Thorsten Spitta (Hrsg.): Beratung in der Softwareentwicklung – Modelle, Methoden, Best Practices

P-99 Andreas Schwill, Carsten Schulte, Marco Thomas (Hrsg.): Didaktik der Informatik

P-100 Peter Forbrig, Günter Siegel, Markus Schneider (Hrsg.): HDI 2006: Hochschuldidaktik der Informatik

P-101 Stefan Böttinger, Ludwig Theuvsen, Susanne Rank, Marlies Morgenstern (Hrsg.): Agrarinformatik im Spannungsfeld zwischen Regionalisierung und globalen Wertschöpfungsketten

P-102 Otto Spaniol (Eds.): Mobile Services and Personalized Environments

P-103 Alfons Kemper, Harald Schöning, Thomas Rose, Matthias Jarke, Thomas Seidl, Christoph Quix, Christoph Brochhaus (Hrsg.): Datenbanksysteme in Business, Technologie und Web (BTW 2007)

P-104 Birgitta König-Ries, Franz Lehner, Rainer Malaka, Can Türker (Hrsg.) MMS 2007: Mobilität und mobile Informationssysteme

P-105 Wolf-Gideon Bleek, Jörg Raasch, Heinz Züllighoven (Hrsg.) Software Engineering 2007

P-106 Wolf-Gideon Bleek, Henning Schwentner, Heinz Züllighoven (Hrsg.) Software Engineering 2007 – Beiträge zu den Workshops

P-107 Heinrich C. Mayr, Dimitris Karagiannis (eds.) Information Systems Technology and its Applications

P-108 Arslan Brömme, Christoph Busch, Detlef Hühnlein (eds.) BIOSIG 2007: Biometrics and Electronic Signatures

P-109 Rainer Koschke, Otthein Herzog, Karl-Heinz Rödiger, Marc Ronthaler (Hrsg.) INFORMATIK 2007 Informatik trifft Logistik Band 1

P-110 Rainer Koschke, Otthein Herzog, Karl-Heinz Rödiger, Marc Ronthaler (Hrsg.) INFORMATIK 2007 Informatik trifft Logistik Band 2

P-111 Christian Eibl, Johannes Magenheim, Sigrid Schubert, Martin Wessner (Hrsg.) DeLFI 2007: 5. e-Learning Fachtagung Informatik

P-112 Sigrid Schubert (Hrsg.) Didaktik der Informatik in Theorie und Praxis

P-113 Sören Auer, Christian Bizer, Claudia Müller, Anna V. Zhdanova (Eds.) The Social Semantic Web 2007 Proceedings of the 1st Conference on Social Semantic Web (CSSW)

P-114 Sandra Frings, Oliver Göbel, Detlef Günther, Hardo G. Hase, Jens Nedon, Dirk Schadt, Arslan Brömme (Eds.) IMF2007 IT-incident management & IT-forensics Proceedings of the 3rd International Conference on IT-Incident Management & IT-Forensics

P-115 Claudia Falter, Alexander Schliep, Joachim Selbig, Martin Vingron and Dirk Walther (Eds.) German conference on bioinformatics GCB 2007

P-116 Witold Abramowicz, Leszek Maciszek (Eds.) Business Process and Services Computing 1st International Working Conference on Business Process and Services Computing BPSC 2007

P-117 Ryszard Kowalczyk (Ed.) Grid service engineering and manegement The 4th International Conference on Grid Service Engineering and Management GSEM 2007

P-118 Andreas Hein, Wilfried Thoben, Hans-Jürgen Appelrath, Peter Jensch (Eds.) European Conference on ehealth 2007

P-119 Manfred Reichert, Stefan Strecker, Klaus Turowski (Eds.) Enterprise Modelling and Information Systems Architectures Concepts and Applications

P-120 Adam Pawlak, Kurt Sandkuhl, Wojciech Cholewa, Leandro Soares Indrusiak (Eds.) Coordination of Collaborative Engineering - State of the Art and Future Challenges

P-121 Korbinian Herrmann, Bernd Bruegge (Hrsg.) Software Engineering 2008 Fachtagung des GI-Fachbereichs Softwaretechnik

P-122 Walid Maalej, Bernd Bruegge (Hrsg.) Software Engineering 2008 - Workshopband Fachtagung des GI-Fachbereichs Softwaretechnik

P-123 Michael H. Breitner, Martin Breunig, Elgar Fleisch, Ley Pousttchi, Klaus Turowski (Hrsg.)
Mobile und Ubiquitäre Informationssysteme – Technologien, Prozesse, Marktfähigkeit
Proceedings zur 3. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2008)

P-124 Wolfgang E. Nagel, Rolf Hoffmann, Andreas Koch (Eds.)
9th Workshop on Parallel Systems and Algorithms (PASA)
Workshop of the GI/ITG Speciel Interest Groups PARS and PARVA

P-125 Rolf A.E. Müller, Hans-H. Sundermeier, Ludwig Theuvsen, Stephanie Schütze, Marlies Morgenstern (Hrsg.)
Unternehmens-IT: Führungsinstrument oder Verwaltungsbürde
Referate der 28. GIL Jahrestagung

P-126 Rainer Gimnich, Uwe Kaiser, Jochen Quante, Andreas Winter (Hrsg.)
10th Workshop Software Reengineering (WSR 2008)

P-127 Thomas Kühne, Wolfgang Reisig, Friedrich Steimann (Hrsg.)
Modellierung 2008

P-128 Ammar Alkassar, Jörg Siekmann (Hrsg.)
Sicherheit 2008
Sicherheit, Schutz und Zuverlässigkeit
Beiträge der 4. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)
2.-4. April 2008
Saarbrücken, Germany

P-129 Wolfgang Hesse, Andreas Oberweis (Eds.)
Sigsand-Europe 2008
Proceedings of the Third AIS SIGSAND European Symposium on Analysis, Design, Use and Societal Impact of Information Systems

P-130 Paul Müller, Bernhard Neumair, Gabi Dreo Rodosek (Hrsg.)
1. DFN-Forum Kommunikations-technologien Beiträge der Fachtagung

P-131 Robert Krimmer, Rüdiger Grimm (Eds.)
3rd International Conference on Electronic Voting 2008
Co-organized by Council of Europe, Gesellschaft für Informatik and E-Voting. CC

P-132 Silke Seehusen, Ulrike Lucke, Stefan Fischer (Hrsg.)
DeLFI 2008:
Die 6. e-Learning Fachtagung Informatik

P-133 Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, Christian Scheideler (Hrsg.)
INFORMATIK 2008
Beherrschbare Systeme – dank Informatik
Band 1

P-134 Heinz-Gerd Hegering, Axel Lehmann, Hans Jürgen Ohlbach, Christian Scheideler (Hrsg.)
INFORMATIK 2008
Beherrschbare Systeme – dank Informatik
Band 2

P-135 Torsten Brinda, Michael Fothe, Peter Hubwieser, Kirsten Schlüter (Hrsg.)
Didaktik der Informatik – Aktuelle Forschungsergebnisse

P-136 Andreas Beyer, Michael Schroeder (Eds.)
German Conference on Bioinformatics GCB 2008

P-137 Arslan Brömme, Christoph Busch, Detlef Hühnlein (Eds.)
BIOSIG 2008: Biometrics and Electronic Signatures

P-138 Barbara Dinter, Robert Winter, Peter Chamoni, Norbert Gronau, Klaus Turowski (Hrsg.)
Synergien durch Integration und Informationslogistik
Proceedings zur DW2008

P-139 Georg Herzwurm, Martin Mikusz (Hrsg.)
Industrialisierung des Software-Managements
Fachtagung des GI-Fachausschusses Management der Anwendungsentwicklung und -wartung im Fachbereich Wirtschaftsinformatik

P-140 Oliver Göbel, Sandra Frings, Detlef Günther, Jens Nedon, Dirk Schadt (Eds.)
IMF 2008 - IT Incident Management & IT Forensics

P-141 Peter Loos, Markus Nüttgens, Klaus Turowski, Dirk Werth (Hrsg.)
Modellierung betrieblicher Informations-systeme (MobIS 2008)
Modellierung zwischen SOA und Compliance Management

P-142 R. Bill, P. Korduan, L. Theuvsen, M. Morgenstern (Hrsg.)
Anforderungen an die Agrarinformatik durch Globalisierung und Klimaveränderung

P-143 Peter Liggesmeyer, Gregor Engels, Jürgen Münch, Jörg Dörr, Norman Riegel (Hrsg.)
Software Engineering 2009
Fachtagung des GI-Fachbereichs Softwaretechnik

P-144 Johann-Christoph Freytag, Thomas Ruf, Wolfgang Lehner, Gottfried Vossen (Hrsg.)
Datenbanksysteme in Business, Technologie und Web (BTW)

P-145 Knut Hinkelmann, Holger Wache (Eds.)
WM2009: 5th Conference on Professional Knowledge Management

P-146 Markus Bick, Martin Breunig, Hagen Höpfner (Hrsg.)
Mobile und Ubiquitäre Informationssysteme – Entwicklung, Implementierung und Anwendung
4. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2009)

P-147 Witold Abramowicz, Leszek Maciaszek, Ryszard Kowalczyk, Andreas Speck (Eds.)
Business Process, Services Computing and Intelligent Service Management
BPSC 2009 · ISM 2009 · YRW-MBP 2009

P-148 Christian Erfurth, Gerald Eichler, Volkmar Schau (Eds.)
9th International Conference on Innovative Internet Community Systems
I²CS 2009

P-149 Paul Müller, Bernhard Neumair, Gabi Dreo Rodosek (Hrsg.)
2. DFN-Forum Kommunikationstechnologien
Beiträge der Fachtagung

P-150 Jürgen Münch, Peter Liggesmeyer (Hrsg.)
Software Engineering
2009 - Workshopband

P-151 Armin Heinzl, Peter Dadam, Stefan Kirn, Peter Lockemann (Eds.)
PRIMIUM
Process Innovation for Enterprise Software

P-152 Jan Mendling, Stefanie Rinderle-Ma, Werner Esswein (Eds.)
Enterprise Modelling and Information Systems Architectures
Proceedings of the 3rd Int'l Workshop EMISA 2009

P-153 Andreas Schwill, Nicolas Apostolopoulos (Hrsg.)
Lernen im Digitalen Zeitalter
DeLFI 2009 – Die 7. E-Learning Fachtagung Informatik

P-154 Stefan Fischer, Erik Maehle Rüdiger Reischuk (Hrsg.)
INFORMATIK 2009
Im Focus das Leben

P-155 Arslan Brömme, Christoph Busch, Detlef Hühnlein (Eds.)
BIOSIG 2009:
Biometrics and Electronic Signatures
Proceedings of the Special Interest Group on Biometrics and Electronic Signatures

P-156 Bernhard Koerber (Hrsg.)
Zukunft braucht Herkunft
25 Jahre »INFOS – Informatik und Schule«

P-157 Ivo Grosse, Steffen Neumann, Stefan Posch, Falk Schreiber, Peter Stadler (Eds.)
German Conference on Bioinformatics 2009

P-158 W. Claupein, L. Theuvsen, A. Kämpf, M. Morgenstern (Hrsg.)
Precision Agriculture Reloaded – Informationsgestützte Landwirtschaft

P-159 Gregor Engels, Markus Luckey, Wilhelm Schäfer (Hrsg.)
Software Engineering 2010

P-160 Gregor Engels, Markus Luckey, Alexander Pretschner, Ralf Reussner (Hrsg.)
Software Engineering 2010 – Workshopband
(inkl. Doktorandensymposium)

P-161 Gregor Engels, Dimitris Karagiannis Heinrich C. Mayr (Hrsg.)
Modellierung 2010

P-162 Maria A. Wimmer, Uwe Brinkhoff, Siegfried Kaiser, Dagmar Lück-Schneider, Erich Schweighofer, Andreas Wiebe (Hrsg.)
Vernetzte IT für einen effektiven Staat
Gemeinsame Fachtagung Verwaltungsinformatik (FTVI) und Fachtagung Rechtsinformatik (FTRI) 2010

P-163 Markus Bick, Stefan Eulgem, Elgar Fleisch, J. Felix Hampe, Birgitta König-Ries, Franz Lehner, Key Pousttchi, Kai Rannenberg (Hrsg.)
Mobile und Ubiquitäre Informationssysteme
Technologien, Anwendungen und Dienste zur Unterstützung von mobiler Kollaboration

P-164 Arslan Brömme, Christoph Busch (Eds.)
BIOSIG 2010: Biometrics and Electronic Signatures Proceedings of the Special Interest Group on Biometrics and Electronic Signatures

P-165 Gerald Eichler, Peter Kropf, Ulrike Lechner, Phayung Meesad, Herwig Unger (Eds.)
10th International Conference on Innovative Internet Community Systems (I^2CS) – Jubilee Edition 2010 –

P-166 Paul Müller, Bernhard Neumair, Gabi Dreo Rodosek (Hrsg.)
3. DFN-Forum Kommunikationstechnologien
Beiträge der Fachtagung

P-167 Robert Krimmer, Rüdiger Grimm (Eds.)
4th International Conference on Electronic Voting 2010
co-organized by the Council of Europe, Gesellschaft für Informatik and E-Voting.CC

P-168 Ira Diethelm, Christina Dörge, Claudia Hildebrandt, Carsten Schulte (Hrsg.)
Didaktik der Informatik
Möglichkeiten empirischer Forschungsmethoden und Perspektiven der Fachdidaktik

P-169 Michael Kerres, Nadine Ojstersek Ulrik Schroeder, Ulrich Hoppe (Hrsg.)
DeLFI 2010 - 8. Tagung der Fachgruppe E-Learning der Gesellschaft für Informatik e.V.

P-170 Felix C. Freiling (Hrsg.)
Sicherheit 2010
Sicherheit, Schutz und Zuverlässigkeit

P-171 Werner Esswein, Klaus Turowski, Martin Juhrisch (Hrsg.)
Modellierung betrieblicher Informationssysteme (MobIS 2010)
Modellgestütztes Management

P-172 Stefan Klink, Agnes Koschmider Marco Mevius, Andreas Oberweis (Hrsg.)
EMISA 2010
Einflussfaktoren auf die Entwicklung flexibler, integrierter Informationssysteme
Beiträge des Workshops der GI-Fachgruppe EMISA (Entwicklungsmethoden für Informationssysteme und deren Anwendung)

P-173 Dietmar Schomburg, Andreas Grote (Eds.)
German Conference on Bioinformatics 2010

P-174 Arslan Brömme, Torsten Eymann, Detlef Hühnlein, Heiko Roßnagel, Paul Schmücker (Hrsg.)
perspeGKtive 2010
Workshop „Innovative und sichere Informationstechnologie für das Gesundheitswesen von morgen“

P-175 Klaus-Peter Fähnrich, Bogdan Franczyk (Hrsg.)
INFORMATIK 2010
Service Science – Neue Perspektiven für die Informatik
Band 1

P-176 Klaus-Peter Fähnrich, Bogdan Franczyk (Hrsg.)
INFORMATIK 2010
Service Science – Neue Perspektiven für die Informatik
Band 2

P-177 Witold Abramowicz, Rainer Alt, Klaus-Peter Fähnrich, Bogdan Franczyk, Leszek A. Maciaszek (Eds.)
INFORMATIK 2010
Business Process and Service Science – Proceedings of ISSS and BPSC

P-178 Wolfram Pietsch, Benedikt Krams (Hrsg.)
Vom Projekt zum Produkt
Fachtagung des GI-Fachausschusses Management der Anwendungsentwicklung und -wartung im Fachbereich Wirtschafts-informatik (WI-MAW), Aachen, 2010

P-179 Stefan Gruner, Bernhard Rumpe (Eds.)
FM+AM`2010
Second International Workshop on Formal Methods and Agile Methods

P-180 Theo Härder, Wolfgang Lehner, Bernhard Mitschang, Harald Schöning, Holger Schwarz (Hrsg.)
Datenbanksysteme für Business, Technologie und Web (BTW)
14. Fachtagung des GI-Fachbereichs „Datenbanken und Informationssysteme“ (DBIS)

P-181 Michael Clasen, Otto Schätzel, Brigitte Theuvsen (Hrsg.)
Qualität und Effizienz durch informationsgestützte Landwirtschaft, Fokus: Moderne Weinwirtschaft

P-182 Ronald Maier (Hrsg.)
6th Conference on Professional Knowledge Management
From Knowledge to Action

P-183 Ralf Reussner, Matthias Grund, Andreas Oberweis, Walter Tichy (Hrsg.)
Software Engineering 2011
Fachtagung des GI-Fachbereichs Softwaretechnik

P-184 Ralf Reussner, Alexander Pretschner, Stefan Jähnichen (Hrsg.)
Software Engineering 2011
Workshopband
(inkl. Doktorandensymposium)

P-185 Hagen Höpfner, Günther Specht, Thomas Ritz, Christian Bunse (Hrsg.)
MMS 2011: Mobile und ubiquitäre Informationssysteme Proceedings zur 6. Konferenz Mobile und Ubiquitäre Informationssysteme (MMS 2011)

P-186 Gerald Eichler, Axel Küpper, Volkmar Schau, Hacène Fouchal, Herwig Unger (Eds.)
11th International Conference on Innovative Internet Community Systems (I^2CS)

P-187 Paul Müller, Bernhard Neumair, Gabi Dreo Rodosek (Hrsg.)
4. DFN-Forum Kommunikations-technologien, Beiträge der Fachtagung 20. Juni bis 21. Juni 2011 Bonn

P-188 Holger Rohland, Andrea Kienle, Steffen Friedrich (Hrsg.)
DeLFI 2011 – Die 9. e-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V. 5.–8. September 2011, Dresden

P-189 Thomas, Marco (Hrsg.)
Informatik in Bildung und Beruf INFOS 2011
14. GI-Fachtagung Informatik und Schule

P-190 Markus Nüttgens, Oliver Thomas, Barbara Weber (Eds.)
Enterprise Modelling and Information Systems Architectures (EMISA 2011)

P-191 Arslan Brömme, Christoph Busch (Eds.)
BIOSIG 2011
International Conference of the Biometrics Special Interest Group

P-192 Hans-Ulrich Heiß, Peter Pepper, Holger Schlingloff, Jörg Schneider (Hrsg.)
INFORMATIK 2011
Informatik schafft Communities

P-193 Wolfgang Lehner, Gunther Piller (Hrsg.)
IMDM 2011

P-194 M. Clasen, G. Fröhlich, H. Bernhardt, K. Hildebrand, B. Theuvsen (Hrsg.)
Informationstechnologie für eine nachhaltige Landbewirtschaftung Fokus Forstwirtschaft

P-195 Neeraj Suri, Michael Waidner (Hrsg.)
Sicherheit 2012
Sicherheit, Schutz und Zuverlässigkeit
Beiträge der 6. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)

P-196 Arslan Brömme, Christoph Busch (Eds.)
BIOSIG 2012
Proceedings of the 11th International Conference of the Biometrics Special Interest Group

P-197 Jörn von Lucke, Christian P. Geiger, Siegfried Kaiser, Erich Schweighofer, Maria A. Wimmer (Hrsg.)
Auf dem Weg zu einer offenen, smarten und vernetzten Verwaltungskultur
Gemeinsame Fachtagung Verwaltungsinformatik (FTVI) und Fachtagung Rechtsinformatik (FTRI) 2012

P-198 Stefan Jähnichen, Axel Küpper, Sahin Albayrak (Hrsg.)
Software Engineering 2012
Fachtagung des GI-Fachbereichs Softwaretechnik

P-199 Stefan Jähnichen, Bernhard Rumpe, Holger Schlingloff (Hrsg.)
Software Engineering 2012
Workshopband

P-200 Gero Mühl, Jan Richling, Andreas Herkersdorf (Hrsg.)
ARCS 2012 Workshops

P-201 Elmar J. Sinz Andy Schürr (Hrsg.)
Modellierung 2012

P-202 Andrea Back, Markus Bick, Martin Breunig, Key Pousttchi, Frédéric Thiesse (Hrsg.)
MMS 2012:Mobile und Ubiquitäre Informationssysteme

P-203 Paul Müller, Bernhard Neumair, Helmut Reiser, Gabi Dreo Rodosek (Hrsg.)
5. DFN-Forum Kommunikations-technologien
Beiträge der Fachtagung

P-204 Gerald Eichler, Leendert W. M. Wienhofen, Anders Kofod-Petersen, Herwig Unger (Eds.)
12th International Conference on Innovative Internet Community Systems (I2CS 2012)

P-205 Manuel J. Kripp, Melanie Volkamer, Rüdiger Grimm (Eds.)
5th International Conference on Electronic Voting 2012 (EVOTE2012)
Co-organized by the Council of Europe, Gesellschaft für Informatik and E-Voting.CC

P-206 Stefanie Rinderle-Ma, Mathias Weske (Hrsg.)
EMISA 2012
Der Mensch im Zentrum der Modellierung

P-207 Jörg Desel, Jörg M. Haake, Christian Spannagel (Hrsg.)
DeLFI 2012: Die 10. e-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V.
24.–26. September 2012

P-208 Ursula Goltz, Marcus Magnor, Hans-Jürgen Appelrath, Herbert Matthies, Wolf-Tilo Balke, Lars Wolf (Hrsg.) INFORMATIK 2012

P-209 Hans Brandt-Pook, André Fleer, Thorsten Spitta, Malte Wattenberg (Hrsg.) Nachhaltiges Software Management

P-210 Erhard Plödereder, Peter Dencker, Herbert Klenk, Hubert B. Keller, Silke Spitzer (Hrsg.) Automotive – Safety & Security 2012 Sicherheit und Zuverlässigkeit für automobile Informationstechnik

P-211 M. Clasen, K. C. Kersebaum, A. Meyer-Aurich, B. Theuvsen (Hrsg.) Massendatenmanagement in der Agrar- und Ernährungswirtschaft Erhebung - Verarbeitung - Nutzung Referate der 33. GIL-Jahrestagung 20. – 21. Februar 2013, Potsdam

P-212 Arslan Brömme, Christoph Busch (Eds.) BIOSIG 2013 Proceedings of the 12th International Conference of the Biometrics Special Interest Group 04.–06. September 2013 Darmstadt, Germany

P-213 Stefan Kowalewski, Bernhard Rumpe (Hrsg.) Software Engineering 2013 Fachtagung des GI-Fachbereichs Softwaretechnik

P-214 Volker Markl, Gunter Saake, Kai-Uwe Sattler, Gregor Hackenbroich, Bernhard Mit schang, Theo Härder, Veit Köppen (Hrsg.) Datenbanksysteme für Business, Technologie und Web (BTW) 2013 13. – 15. März 2013, Magdeburg

P-215 Stefan Wagner, Horst Lichter (Hrsg.) Software Engineering 2013 Workshopband (inkl. Doktorandensymposium) 26. Februar – 1. März 2013, Aachen

P-216 Gunter Saake, Andreas Henrich, Wolfgang Lehner, Thomas Neumann, Veit Köppen (Hrsg.) Datenbanksysteme für Business, Technologie und Web (BTW) 2013 – Workshopband 11. – 12. März 2013, Magdeburg

P-217 Paul Müller, Bernhard Neumair, Helmut Reiser, Gabi Dreo Rodosek (Hrsg.) 6. DFN-Forum Kommunikations-technologien Beiträge der Fachtagung 03.–04. Juni 2013, Erlangen

P-218 Andreas Breiter, Christoph Rensing (Hrsg.) DeLFI 2013: Die 11 e-Learning Fachtagung Informatik der Gesellschaft für Informatik e.V. (GI) 8. – 11. September 2013, Bremen

P-219 Norbert Breier, Peer Stechert, Thomas Wilke (Hrsg.) Informatik erweitert Horizonte INFOS 2013 15. GI-Fachtagung Informatik und Schule 26. – 28. September 2013

P-220 Matthias Horbach (Hrsg.) INFORMATIK 2013 Informatik angepasst an Mensch, Organisation und Umwelt 16. – 20. September 2013, Koblenz

P-221 Maria A. Wimmer, Marijn Janssen, Ann Macintosh, Hans Jochen Scholl, Efthimios Tambouris (Eds.) Electronic Government and Electronic Participation Joint Proceedings of Ongoing Research of IFIP EGOV and IFIP ePart 2013 16. – 19. September 2013, Koblenz

P-222 Reinhard Jung, Manfred Reichert (Eds.) Enterprise Modelling and Information Systems Architectures (EMISA 2013) St. Gallen, Switzerland September 5. – 6. 2013

P-223 Detlef Hühnlein, Heiko Roßnagel (Hrsg.) Open Identity Summit 2013 10. – 11. September 2013 Kloster Banz, Germany

P-224 Eckhart Hanser, Martin Mikusz, Masud Fazal-Baqaie (Hrsg.) Vorgehensmodelle 2013 Vorgehensmodelle – Anspruch und Wirklichkeit 20. Tagung der Fachgruppe Vorgehensmodelle im Fachgebiet Wirtschaftsinformatik (WI-VM) der Gesellschaft für Informatik e.V. Lörrach, 2013

P-225 Hans-Georg Fill, Dimitris Karagiannis, Ulrich Reimer (Hrsg.) Modellierung 2014 19. – 21. März 2014, Wien

P-226 M. Clasen, M. Hamer, S. Lehnert, B. Petersen, B. Theuvsen (Hrsg.) IT-Standards in der Agrar- und Ernährungswirtschaft Fokus: Risiko- und Krisenmanagement Referate der 34. GIL-Jahrestagung 24. – 25. Februar 2014, Bonn

P-227 Wilhelm Hasselbring,
Nils Christian Ehmke (Hrsg.)
Software Engineering 2014
Fachtagung des GI-Fachbereichs Softwaretechnik
25. – 28. Februar 2014
Kiel, Deutschland

P-228 Stefan Katzenbeisser, Volkmar Lotz, Edgar Weippl (Hrsg.)
Sicherheit 2014
Sicherheit, Schutz und Zuverlässigkeit
Beiträge der 7. Jahrestagung des Fachbereichs Sicherheit der Gesellschaft für Informatik e.V. (GI)
19. – 21. März 2014, Wien

P-230 Arslan Brömme, Christoph Busch (Eds.)
BIOSIG 2014
Proceedings of the 13th International Conference of the Biometrics Special Interest Group
10. – 12. September 2014 in Darmstadt, Germany

P-231 Paul Müller, Bernhard Neumair, Helmut Reiser, Gabi Dreo Rodosek (Hrsg.)
7. DFN-Forum Kommunikationstechnologien
16. – 17. Juni 2014
Fulda

P-232 E. Plödereder, L. Grunske, E. Schneider, D. Ull (Hrsg.)
INFORMATIK 2014
Big Data – Komplexität meistern
22. – 26. September 2014
Stuttgart

P-233 Stephan Trahasch, Rolf Plötzner, Gerhard Schneider, Claudia Gayer, Daniel Sassiat, Nicole Wöhrle (Hrsg.)
DeLFI 2014 – Die 12. e-Learning Fachtagung Informatik
der Gesellschaft für Informatik e.V.
15. – 17. September 2014
Freiburg

P-234 Fernand Feltz, Bela Mutschler, Benoît Otjacques (Eds.)
Enterprise Modelling and Information Systems Architectures
(EMISA 2014)
Luxembourg, September 25-26, 2014

P-235 Robert Giegerich,
Ralf Hofestädt,
Tim W. Nattkemper (Eds.)
German Conference on Bioinformatics 2014
September 28 – October 1
Bielefeld, Germany

P-236 Martin Engstler, Eckhart Hanser, Martin Mikusz, Georg Herzwurm (Hrsg.)
Projektmanagement und Vorgehensmodelle 2014
Soziale Aspekte und Standardisierung
Gemeinsame Tagung der Fachgruppen Projektmanagement (WI-PM) und Vorgehensmodelle (WI-VM) im Fachgebiet Wirtschaftsinformatik der Gesellschaft für Informatik e.V., Stuttgart 2014

P-237 Detlef Hühnlein, Heiko Roßnagel (Hrsg.)
Open Identity Summit 2014
4.–6. November 2014
Stuttgart, Germany

P-238 Arno Ruckelshausen, Hans-Peter Schwarz, Brigitte Theuvsen (Hrsg.)
Informatik in der Land-, Forst- und Ernährungswirtschaft
Referate der 35. GIL-Jahrestagung
23. – 24. Februar 2015, Geisenheim

P-239 Uwe Aßmann, Birgit Demuth, Thorsten Spitta, Georg Püschel, Ronny Kaiser (Hrsg.)
Software Engineering & Management 2015
17.-20. März 2015, Dresden

P-240 Herbert Klenk, Hubert B. Keller, Erhard Plödereder, Peter Dencker (Hrsg.)
Automotive – Safety & Security 2015
Sicherheit und Zuverlässigkeit für automobile Informationstechnik
21.–22. April 2015, Stuttgart

P-241 Thomas Seidl, Norbert Ritter, Harald Schöning, Kai-Uwe Sattler, Theo Härder, Steffen Friedrich, Wolfram Wingerath (Hrsg.)
Datenbanksysteme für Business, Technologie und Web (BTW 2015)
04. – 06. März 2015, Hamburg

P-242 Norbert Ritter, Andreas Henrich, Wolfgang Lehner, Andreas Thor, Steffen Friedrich, Wolfram Wingerath (Hrsg.)
Datenbanksysteme für Business, Technologie und Web (BTW 2015) – Workshopband
02. – 03. März 2015, Hamburg

P-243 Paul Müller, Bernhard Neumair, Helmut Reiser, Gabi Dreo Rodosek (Hrsg.)
8. DFN-Forum Kommunikationstechnologien
06.–09. Juni 2015, Lübeck

P-244 Alfred Zimmermann,
Alexander Rossmann (Eds.)
Digital Enterprise Computing
(DEC 2015)
Böblingen, Germany June 25-26, 2015

P-245 Arslan Brömme, Christoph Busch ,
Christian Rathgeb, Andreas Uhl (Eds.)
BIOSIG 2015
Proceedings of the 14th International
Conference of the Biometrics Special
Interest Group
09.–11. September 2015
Darmstadt, Germany

P-246 Douglas W. Cunningham, Petra Hofstedt,
Klaus Meer, Ingo Schmitt (Hrsg.)
INFORMATIK 2015
28.9.-2.10. 2015, Cottbus

P-247 Hans Pongratz, Reinhard Keil (Hrsg.)
DeLFI 2015 – Die 13. E-Learning
Fachtagung Informatik der Gesellschaft
für Informatik e.V. (GI)
1.–4. September 2015
München

P-248 Jens Kolb, Henrik Leopold, Jan Mendling
(Eds.)
Enterprise Modelling and Information
Systems Architectures
Proceedings of the 6th Int. Workshop on
Enterprise Modelling and Information
Systems Architectures, Innsbruck, Austria
September 3-4, 2015

P-249 Jens Gallenbacher (Hrsg.)
Informatik
allgemeinbildend begreifen
INFOS 2015 16. GI-Fachtagung
Informatik und Schule
20.–23. September 2015

P-250 Martin Engstler, Masud Fazal-Baqaie,
Eckhart Hanser, Martin Mikusz,
Alexander Volland (Hrsg.)
Projektmanagement und
Vorgehensmodelle 2015
Hybride Projektstrukturen erfolgreich
umsetzen
Gemeinsame Tagung der Fachgruppen
Projektmanagement (WI-PM) und
Vorgehensmodelle (WI-VM) im
Fachgebiet Wirtschaftsinformatik
der Gesellschaft für Informatik e.V.,
Elmshorn 2015

P-251 Detlef Hühnlein, Heiko Roßnagel,
Raik Kuhlisch, Jan Ziesing (Eds.)
Open Identity Summit 2015
10.–11. November 2015
Berlin, Germany

P-252 Jens Knoop, Uwe Zdun (Hrsg.)
Software Engineering 2016
Fachtagung des GI-Fachbereichs
Softwaretechnik
23.–26. Februar 2016, Wien

P-253 A. Ruckelshausen, A. Meyer-Aurich,
T. Rath, G. Recke, B. Theuvsen (Hrsg.)
Informatik in der Land-, Forst- und
Ernährungswirtschaft
Fokus: Intelligente Systeme – Stand der
Technik und neue Möglichkeiten
Referate der 36. GIL-Jahrestagung
22.-23. Februar 2016, Osnabrück

P-254 Andreas Oberweis, Ralf Reussner (Hrsg.)
Modellierung 2016
2.–4. März 2016, Karlsruhe

P-255 Stefanie Betz, Ulrich Reimer (Hrsg.)
Modellierung 2016 Workshopband
2.–4. März 2016, Karlsruhe

P-256 Michael Meier, Delphine Reinhardt,
Steffen Wendzel (Hrsg.)
Sicherheit 2016
Sicherheit, Schutz und Zuverlässigkeit
Beiträge der 8. Jahrestagung des
Fachbereichs Sicherheit der
Gesellschaft für Informatik e.V. (GI)
5.–7. April 2016, Bonn

The titles can be purchased at:

Köllen Druck + Verlag GmbH
Ernst-Robert-Curtius-Str. 14 · D-53117 Bonn
Fax: +49 (0)228/9898222
E-Mail: druckverlag@koellen.de