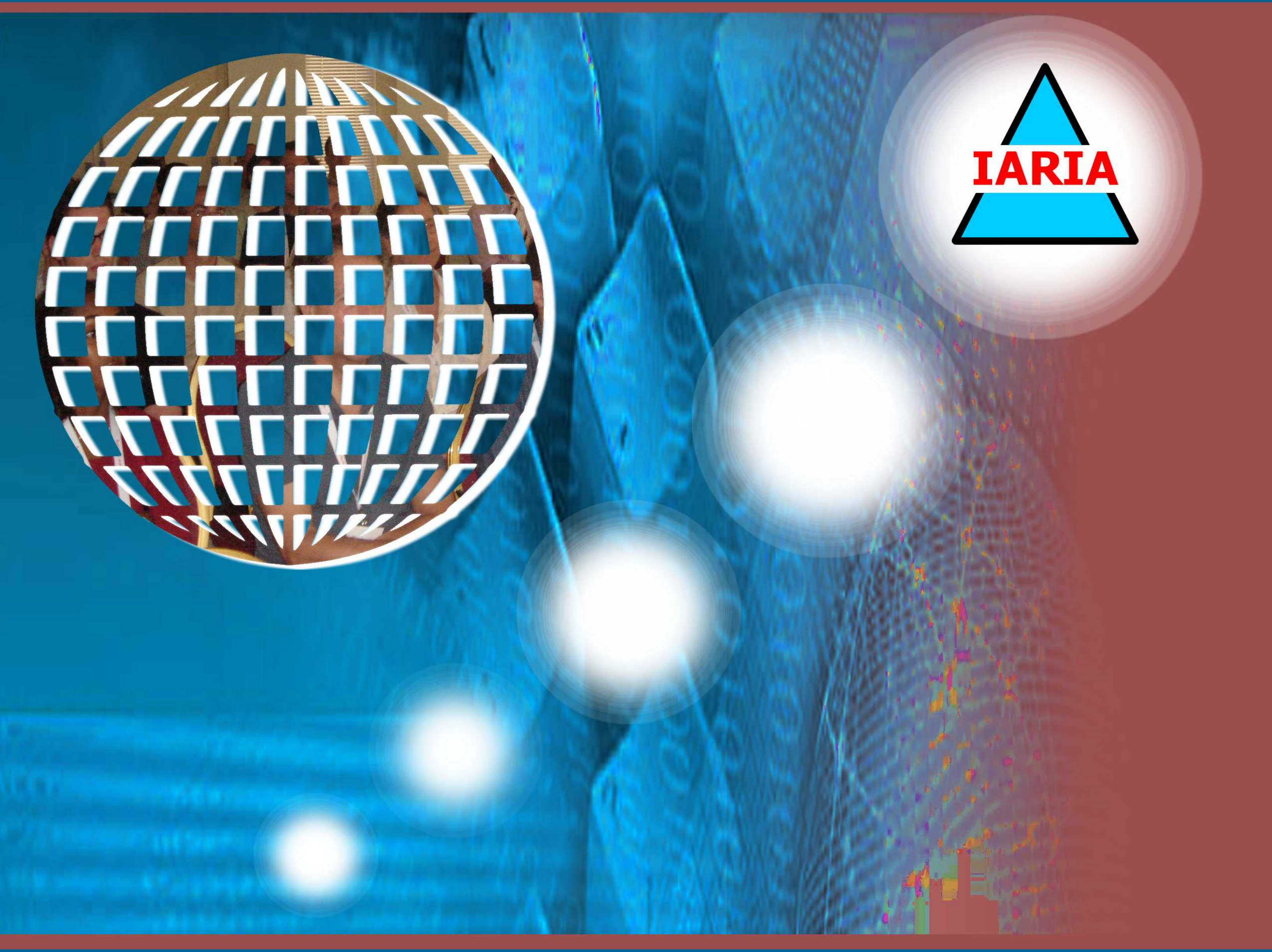
International Journal on Advances in Security



2014 vol. 7 nr. 3&4

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

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Security
Volume 7, Number 3 & 4, 2014

Rivalino Matias Jr ., Federal University of Uberlandia, Brazil
Manuel Mazzara, UNU-IIST, Macau / Newcastle University, UK
Carla Merkle Westphall, Federal University of Santa Catarina (UFSC), Brazil
Ajaz H. Mir, National Institute of Technology, Srinagar, India
Jose Manuel Moya, Technical University of Madrid, Spain
Leonardo Mostarda, Middlesex University, UK
Jogesh K. Muppala, The Hong Kong University of Science and Technology, Hong Kong
Syed Naqvi, CETIC (Centre d'Excellence en Technologies de l'Information et de la Communication),Belgium
Sarmistha Neogy, Jadavpur University, India
Mats Neovius, Åbo Akademi University, Finland
Jason R.C. Nurse, University of Oxford, UK
Peter Parycek, Donau-Universität Krems, Austria
Konstantinos Patsakis, Rovira i Virgili University, Spain
João Paulo Barraca, University of Aveiro, Portugal
Sergio Pozo Hidalgo, University of Seville, Spain
Vladimir Privman, Clarkson University, USA
Yong Man Ro, KAIST (Korea advanced Institute of Science and Technology), Korea
Rodrigo Roman Castro, Institute for Infocomm Research (Member of A*STAR), Singapore
Heiko Roßnagel, Fraunhofer Institute for Industrial Engineering IAO, Germany
Claus-Peter Rückemann, Leibniz Universität Hannover / Westfälische Wilhelms-Universität Münster / North-German Supercomputing Alliance, Germany
Antonio Ruiz Martinez, University of Murcia, Spain
Paul Sant, University of Bedfordshire, UK
Reijo Savola, VTT Technical Research Centre of Finland, Finland
Peter Schartner, University of Klagenfurt, Austria
Alireza Shameli Sendi, Ecole Polytechnique de Montreal, Canada
Dimitrios Serpanos, Univ. of Patras and ISI/RC ATHENA, Greece
Pedro Sousa, University of Minho, Portugal
George Spanoudakis, City University London, UK
Lars Strand, Nofas, Norway
Young-Joo Suh, Pohang University of Science and Technology (POSTECH), Korea
Jani Suomalainen, VTT Technical Research Centre of Finland, Finland
Enrico Thomae, Ruhr-University Bochum, Germany
Tony Thomas, Indian Institute of Information Technology and Management - Kerala, India
Panagiotis Trimintzios, ENISA, EU
Peter Tröger, Hasso Plattner Institute, University of Potsdam, Germany
Simon Tsang, Applied Communication Sciences, USA
Marco Vallini, Politecnico di Torino, Italy
Bruno Vavala, Carnegie Mellon University, USA
Mthulisi Velempini, North-West University, South Africa
Miroslav Velev, Aries Design Automation, USA
Salvador E. Venegas-Andraca, Tecnológico de Monterrey / Texia, SA de CV, Mexico
Szu-Chi Wang, National Cheng Kung University, Tainan City, Taiwan R.O.C.
Piyi Yang, University of Shanghai for Science and Technology, P. R. China
Rong Yang, Western Kentucky University , USA
Hee Yong Youn, Sungkyunkwan University, Korea

Bruno Bogaz Zarpelao, State University of Londrina (UEL), Brazil
Wenbing Zhao, Cleveland State University, USA

International Journal on Advances in Security
Volume 7, Numbers 3 & 4, 2014

CONTENTS

David Musliner, SIFT, LLC, USA
Jeffrey Rye, SIFT, LLC, USA

A Secure Logging Framework with Focus on Compliance

Felix von Eye, David Schmitz, and Wolfgang Hommel
Leibniz Supercomputing Centre, Munich Network Management Team, Garching n. Munich, Germany
Email:{voneye,schmitz,hommel}@lrz.de

***Abstract*—Handling log messages securely, for example, on servers or embedded devices, has often relied on cryptographic messages authentication codes (MACs) to ensure log file integrity: Any modification or deletion of a log entry will invalidate the MAC, making the tampering evident. However, organizational security requirements regarding log files have changed significantly over the decades. For example, European privacy and personal data protection laws mandate that certain information, such as IP (internet protocol) addresses, must only be stored for a certain retention period, typically seven days. Traditional log file security measures, however, do not support the delayed deletion of partial log message information for such compliance reasons. This article presents SLOPPI (secure logging with privacy protection and integrity), a three-tiered log management framework with focus on integrity management and compliance as well as optional support for encryption-based confidentiality of log messages.**

Keywords-log file management; secure logging; compliance; log message encryption; privacy by design.

I. Introduction

For the secure logging, von Eye et al. presented SLOPPI [1] – a framework for secure logging with privacy protection and integrity, which is extended in this article. This framework helps to ensure that the log files, independent of the storage format, fulfill the well-known goals of information technology security:

- The log's *integrity* must be ensured: Neither a malicious administrator nor an attacker, who successfully has compromised a system, shall be able to delete or modify existing, or insert bogus log entries.
- The log shall not violate *compliance* criteria. For example, European data protection laws regulate the retention of personal data, which includes, among many others, user names and IP addresses. These restrictions also apply to log entries according to several German courts' verdicts that motivate the presented approach; details are part of previous work [2].
- The *confidentiality* of log entries shall be safeguarded; i. e., read access to log entries shall be confined to an arbitrary set of users.
- The *availability* of log entries shall be made sure of.

The security of log files is a very important aspect in the overall security concept of a service or device. Many attacks or resource abuse cases can be detected by analyzing log files as part of a forensics process, just like system or service breakdowns. With the knowledge embedded in log data, administrators and forensics experts are often able to reconstruct the way an attacker was intruding the system or the root cause of the system disaster.

Because of the concentrated information, the log data is often a primary target of attackers once they have compromised the system. On the one hand, an attacker could erase the whole log file to cover up the traces. This a very efficient way but it also provides a clear information that something went wrong on the system, which most likely will arouse the system administrator's suspicion or trigger an automated alert. When this happens, the administrator is able to detect the attack very fast, even if not every detail can be reconstructed. On the other hand, an attacker could change some of the log entries in a way that the manipulation is not obvious. The approach presented in this article focuses on the second scenario. In any way it would be possible for an attacker to fully delete the log files, which cannot be circumvented as long as the log file resides on a fully compromised machine. Even if there is the possibility to store the log files on an external system, such as a log server, the attacker can be able to disturb the connection between the system and the log server, e. g., by firewalling the connection, once the system has been hacked and the attacker managed to get administrator privileges. But beside this, there are also a lot of systems, which cannot be connected to a central log server, e. g., because of the mobility of the systems or when the organization is too small to operate a dedicated central log server.

This motivates the SLOPPI approach [1], which allows administrators to protect their log files against unwanted changes, while the deletion of log files, e. g., a regular log rotation, is still possible. The SLOPPI architecture allows administrators to have long time logs, while privacy related parts of the log file can be deleted after a well-defined period of time. In this article, the SLOPPI approach, which has substantially been improved since its introduction in [1], is presented. The primary limitation of the previous SLOPPI approach was, that the deletion of log entries also deletes any other information inside this log entry. So, it was not possible to keep parts of the information, e. g., for statistical or diagnostic reasons. In this paper we deal with this drawback and present an improved and more detailed approach, which is now able to keep some predefined parts of the information.

This work is motivated by the large-scale distributed environment of the SASER-SIEGFRIED project (Safe and Secure European Routing) [3], in which more than 50 project partners design and implement network architectures and technologies for secure future networks. The project's goal is to remedy security vulnerabilities of today's IP layer networks in the 2020 timeframe. Thereby, security mechanisms for future networks are designed based on an analysis of the currently predominant security problems on the IP layer, as well as upcoming issues such as vendor-created loopholes and SDN-based (software defined network) traffic anomaly detection. The project focuses on inter-domain routing, and routing decisions are based on security metrics that are part of log entries sent by active network components to central network management systems; therefore, the integrity of this data must be protected, providing a use case that is similar to traditional intra-organizational log file management applications.

The remainder of this article is structured as follows: Section II introduces the terminology and notation that is used throughout the article. In Section III, the related work and state of the art as well as its influence on the design of SLOPPI are discussed. SLOPPI's architecture and workflows are presented in-depth in Section IV. The process for verifying the integrity of SLOPPI log files is specified in Section V. Before the article's conclusion, Section VI analyzes the security properties of SLOPPI.

II. Terminology

In this article, a few terms and symbols are used to avoid ambiguity. These symbols and terms have the following meaning:

- In the special focus of this work is the untrusted device $\mathcal{U}$, which could be for example a web server or a Linux system. As a matter of its regular operations, $\mathcal{U}$ produces log data, which is saved in one or more log files. As $\mathcal{U}$ is a system, which is not necessarily hardened in any matter, it can be assumed that $\mathcal{U}$ may be compromised by an attacker and therefore, the log data is not guaranteed to be *trustworthy*, i. e., the security goals confidentiality, integrity, and availability cannot reliably be achieved. However, the SLOPPI approach can be used to ensure the integrity and compliance of log data produced by $\mathcal{U}$, making it *reliable* under this specific aspect.
- A trusted machine $\mathcal{T}$. In any related work there is a need for a separate machine $\mathcal{T} \neq \mathcal{U}$. The working assumption in the related work is that $\mathcal{T}$ is secure, trustworthy, and not under the control of an attacker at any point in time. In the presented approach, $\mathcal{T}$ is not needed any more as a fully-fledged computer system. To ensure a uniform notation, $\mathcal{T}$ is also used in the following sections in the meaning of a trusted storage for a security key, e. g., a piece of paper that is written on with a pencil. As long as this written paper cannot be read by an outside attacker, it can be assumed as trusted enough. Certainly there are also other solutions, for example, saving the key on a USB memory device (universal serial bus), but in this article the focus is not on the hardening of $\mathcal{T}$ because offline and analogous solutions are sufficient.
- The verifier $\mathcal{V}$. Related work often differentiates $\mathcal{T}$ and $\mathcal{V}$; $\mathcal{V}$ then is only responsible for verifying the integrity and compliance of a log file or log stream. In this case, $\mathcal{T}$ is only used to store the needed keys and $\mathcal{V}$ does not have to be as trustworthy as $\mathcal{T}$. Also in this case, $\mathcal{T}$ is able to modify any log entry, while $\mathcal{V}$ is not.

These symbols are used for cryptographic operations:

- A strong cryptographic hash function H, which has to be a one way function, i. e., a function, which is easy to compute but hard to reverse, e. g., `SHA-256`(m) (Secure Hash Algorithm) or Keccak.
- $\text{HMAC}_k(m)$ (hash-based message authentication code). The message authentication code of the message m using the key k.

SLOPPI does not anticipate, which particular function should be used for cryptographic functions; instead, they should be chosen specifically based on each implementation's security requirements and constraints, such as available processing power, system and data sensitivity, and induced storage overhead.

Furthermore, without loss of generality, the terms *log files*, *log entries*, and *log messages* are discerned as follows:

- A *log file* is an ordered set of *log entries*. The order is implied by the order, in which log messages are received by SLOPPI.
- In line-based logs, a *log entry* normally corresponds to one line of the log file. Otherwise, a log entry consists of all information, which is related to one event. For example, on a typical Linux system, the file `/var/log/messages` is a text-based log file and each line therein is a log entry; log entries are written in chronological order to this log file. For massively parallel operations, the resulting order is determined by the implementation of a syslog-style system service at run-time based on criteria outside the scope of this article. Log entries cannot be re-ordered once they have been written to a log file in the SLOPPI architecture, which is consistent with related work. Other log file formats, such as the proprietary binary Microsoft Windows event log format, can also be used with SLOPPI; for simplicity, however, all the examples given in this article refer to text-based log files with one log entry per line of text.
- *Log messages* are the payload of *log entries*; typically, log messages are human-readable character strings that are created by applications or system/device processes. Besides a log message, a log entry includes metadata, such as a timestamp and information about the log message source. Log messages typically have an application-specific structure of their own, which SLOPPI exploits for its slicing technique as detailed below.

As shown in Figure 4, the SLOPPI approach uses several log files, which are related to each other in the following way: The *master log* L_m is the root of the SLOPPI data structure and only used twice a day to first generate and to then close a new integrity stream for the so-called *daily log* L_d. This log file L_d is basically used to minimize the storage needs for the master log L_m, which must never be deleted and therefore shall not contain any personal data or otherwise potentially compliance-offending content; otherwise, no complete verification of the integrity of all log messages could be performed. L_d is kept as long as necessary and contains a new integrity stream for the *application logs* L_a. These logs, e. g., the `access.log` or the `error.log` generated by an Apache web server or the `firewall.log` created by a local firewall, are restarted from scratch once per day and yield all information generated by the related processes; they are extended by integrity check data. Please note that other intervals than 24 hours are arbitrarily possible, but daily log file rotation are most commonly used and the term is used here for its clarity.

After an arbitrarily specified retention time – seven days in the SASER scenario – these logs, which contain privacy-law protected data, have to be cleaned up or purged completely because of legal constraints in Germany and various other countries. It is important to mention that a simple deletion of whole log files or log entries would also remove any information about attempted intrusions and other attack sources. This would cover an intruder, who could be detected by analyzing the log files, so the log file should be analyzed periodically before this automated deletion. Other time periods than full days or a rotation that is based, e. g., on a maximum number of log entries per log file, as well as other deletion periods may be applied – but for the sake of simplicity *daily* logs and a seven day deletion period are used for the remainder of this article. This setup is also used in the SASER project and currently recommended for IT services operated by European providers.

Extending [1], this article presents details about how the *application logs* can be handled with a fine-grained policy that allows to keep as much information as needed arbitrarily longer than the mentioned seven days, while still being compliant. We also discuss log message encryption to ensure confidentiality and how SLOPPI can be used to secure structured log messages, for example, in order to remove substrings from log messages for privacy reasons, to serve as a data source for business intelligence tools, and to facilitate the visualization of security-related log entries.

III. Related work

With the exception of Section III-A, none of related work offers a possibility to fulfill compliance as it is not possible to delete log entries or parts thereof a posteriori. Complementary, the approach summarized in Section III-A does not address the integrity issues.

A. Privacy-enhancing log rotation

Metzger et al. presented an organization-wide concept for privacy-enhancing log rotation in [4]. In this work, log entries are deleted by log file rotation after a period of seven days, which is a common retention period in Germany based on several privacy-related verdicts. Based on surveys, Metzger et al. identified more than 200 different types of log entry sources that contain personal information in a typical higher education data center. Although deleting log entries after seven days seems to be a simple solution, the authors discuss the challenges of implementing and enforcing a strict data retention policy in large-scale distributed environments.

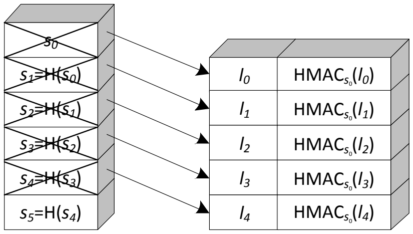


Fig. 1. The basic idea behind Forward Integrity as suggested in [5].

B. Forward integrity

Bellare and Yee introduced the term *Forward Integrity* in [5]. This approach is based on the combination of log entries with message authentication codes (MACs). Once a new log file is started, a secret s_0 is generated on $\mathcal{U}$, which has to be sent in a secure way to a trusted $\mathcal{T}$. This secret is necessary to verify the integrity of a log file.

Once the first log entry l_0 is written to the log file, the HMAC of l_0 based on the key s_0 is calculated and also written to the log file. To protect the secret s_0, there is another calculation of $s_1 = H(s_0)$, which is the new secret for the next log entry l_1. To prevent that an attacker can easily create or modify log entries, the old and already used secret key for the MAC function is erased after the calculation securely. Because of the characteristics of one-way functions, it is not possible for an attacker to derive the previous key backwards in maintainable time. Figure 1 shows the underlying idea.

In their approach, in order to verify the integrity of the log file, $\mathcal{V}$ has to know the initial key to verify all entries in sequential order. If the log entry and the MAC do not correspond, the log file has been corrupted from this moment on, and any subsequent entry is no longer trustworthy.

However, the strict use of forward integrity also prohibits authorized changes to log entries; for example, if personal data shall be removed from log entries after seven days, the old MAC must be thrown away and a new MAC has to be calculated. While this is not a big issue from a computational complexity perspective, it means that the integrity of old log entries may be violated during this rollover if $\mathcal{U}$ has meanwhile been compromised.

C. Encrypted log files

Schneier and Kelsey developed a cryptographic scheme to secure encrypted log files in [6]. They motivated the approach for encrypting each log entry with the need of confidential logging, e. g., in financial applications. Figure 2 shows the process to save a new log entry. Any log entry D_j on $\mathcal{U}$ is encrypted with the key K_j, which in turn is built from the secret A_j (in this article s_j) and an entry type W_j. This entry type allows $\mathcal{V}$ to only verify predefined log entries. There is also some more information stored in a log entry, namely Y_j

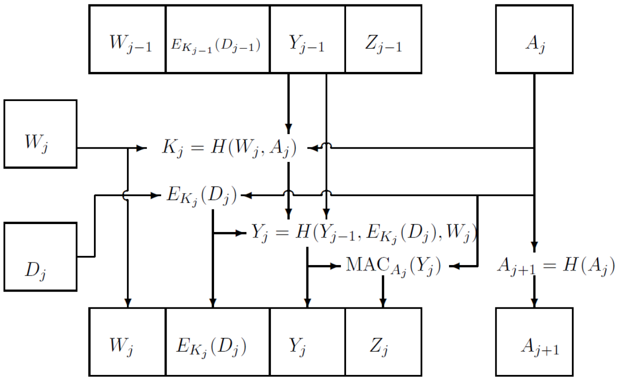


Fig. 2. The Schneier and Kelsey approach taken from [6].

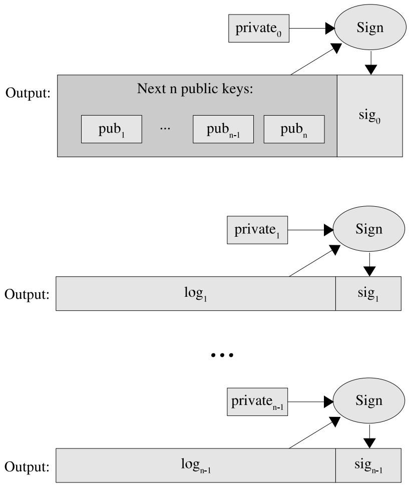


Fig. 3. The Holt approach taken from [7].

and Z_j, which are used to allow the verification of a log entry without the need of decryption of D_j. Therefore, only $\mathcal{T}$ is able to modify the log files.

However, this approach does not allow for the deletion method of log entries or parts thereof because then the verification would inevitably break.

D. Public key encryption

Holt used a public/private-key-based verification process in his approach to allow a complete disjunction of $\mathcal{T}$ and $\mathcal{V}$ in [7]. Therefore, a limited amount of public/private-key pairs are generated. The public keys are stored in a meta-log entry, which is signed with the first public key, which should be erased afterwards securely. All other log entries are also signed with the precomputed private keys. If there are no more keys left, a new limited amount of public/private-key pairs are generated.

The main benefit of this approach is that the verifier $\mathcal{V}$ cannot modify any log entry because it only knows the public keys, which can be used for verifying the signature but does not allow any inference on the used private key.

E. Aggregated Signatures

In scenarios where disk space is the limiting factor it is necessary that the signature, which protects each single log entry, does not take much space. In all of the approaches sketched above, the disk usage by signatures is within $O(n)$, where n is the total amount of log entries. To deal with a more space-constrained scenario, Ma and Tsudik presented a new signature scheme, which aggregates all signatures of the log entries in [8]. This approach uses archiving so that the necessary disk space amount is reduced to only $O(1)$.

The main drawback of this approach is that a manipulation of a single log entry would break the verification process, yet the verifier is not able to determine, which (presumably modified) entry causes the verification process to fail. As a consequence, it is also not possible to delete or to modify log entries, e. g., to remove personal data after reaching the maximum retention time.

F. BAF

Yavuz and Ning specified how log entries can be secured by using blind signatures in [9]. Their approach uses the log entry combined with the actual number of the log entry. For example, if the log entry D_n is the nth entry in the log file, D_n is combined with the number n to prevent an attacker from reordering the log entries. This result is hashed and modified with the secret key (a, b) by using a simple addition and multiplication modulo a large prime p. As in all other approaches, the secret key is updated and the previous version securely deleted.

The most interesting result of this approach is that a verification is possible for a verifier $\mathcal{V}$, while it is not possible for $\mathcal{V}$ to modify any entry in the log file. This property is normally only satisfied by public/private key schemes, which are typically very expensive to compute.

IV. The SLOPPI architecture and workflows

SLOPPI, as presented in [1], follows the classic client-server architecture of well-known POSIX (Portable Operating System Interface) logging mechanisms, such as syslog-ng and rsyslog. Any SLOPPI implementation therefore is a continuously running process, which offers interfaces, such as an application programming interface (API) and IPC (IP code), TCP (Transmission Control Protocol) or UDP (User Datagram Protocol) sockets, to receive new log messages from various system services, applications, or remote servers. After internal processing, log entries are stored in plain-text files in the local file system, where they can be processed with whichever log file viewing mechanism the local users are familiar with; alternatively, log entries can be forwarded to remote SLOPPI servers where they are treated in the same manner. If the application log files make use of the optional encryption, SLOPPI tools can be used to decrypt the log entries using standard input and output channels, typically in combination with POSIX pipes. Similar tools can be used to strip any SLOPPI-specific information from log entries so any other log file processing tools can be used for parsing and processing the application log files even if they are not aware of the extensions brought by the SLOPPI data format.

As SLOPPI has been designed with compliance regulations as its primary motivation, ensuring integrity and allowing for log-file rotation without cryptographic re-keying are its core functionality. In the following sections, first described is how the master log, the daily log, and the application log are intertwined to achieve these properties. Afterwards follows a discussion of the various optional functionalities for the application logs, e. g., the encryption of application log messages.

A. The SLOPPI log file hierarchy

The analysis of the related work shows that there is no solution yet that fulfills both necessary minimum requirements for log files: *integrity* and *compliance*, where the latter requires making changes to integrity-checked log entries once they have reached a certain age. The SLOPPI approach combines key operations from previous approaches in a new innovative way to achieve both characteristics. As introduced in Section II, a couple of types of log files, which are all handled a bit differently, are used for the framework. They form the following hierarchy as shown in Figure 4:

- The master log file is the root of the SLOPPI data format. It is created only once and must not be deleted. If it is deleted, e. g., by an attacker, integrity checking is no longer possible.
- The daily log files are, as implied by their name, created in a daily manner. Although other rollover periods could be used, such as hourly or weekly, we refer to them as daily logs for the sake of simplicity and because they are a de-facto industry standard. Daily log files can be deleted after a retention period, for which 7 days is the standard setting; it is, however, recommended to delete the affected application log files first.
- Application log files are the only log files, in which actual payload log messages are stored – both the master log and the daily logs only contain SLOPPI-specific meta-information. There can be an arbitrary number of application log files depending on how many files all the log information should be scattered across. SLOPPI supports the typical syslog-like distribution of log messages to log files based on the originating host, application, log level, and log message content in an administrator-configured manner. While storing a log message in exactly one log file is the usual mode of operation, the same log message can be logged to multiple log files if desired, or thrown away without being written to a file, and therefore without influencing the integrity mechanism. As an alternative to local log files, log entries can also be forwarded to remote SLOPPI services, which treat them similarly to locally logged messages; communication is secured using TLS (Transport Layer Security) connections.

At the very core of SLOPPI, the master log file L_m has to be secured. As it has only a few entries per day, it can be protected using a public key scheme, e. g., RSA, to protect the log entries, which is described in detail in the upcoming Section IV-B. In the subsequent sections, the two keys of a public key scheme are called *signing key* (k_{sign}) and *authentication key* (k_{auth}).

Then the daily log file L_d is considered in Section IV-C. Similar to the master log file, it only has very few entries per day, but they already must be considered too many entries for using public key schemes, so a symmetric key scheme is certainly the best choice. Finally, application log file details and options are presented in Section IV-D.

B. The SLOPPI Master Log File

As stated above, the master log file L_m contains important data of the SLOPPI approach to protect the integrity of the log files. To protect L_m, the following steps are necessary:

1) Log Initialization: Whenever a new master log is initialized, $\mathcal{U}$ generates an authentication key (k^1_{auth}) and a signing key (k^1_{sign}). These two keys are important to protect (using k^1_{sign}) now and to verify (using k^1_{auth}) the log file later. As the verification key should not be stored on $\mathcal{U}$, it is sent to $\mathcal{T}$ over a secure connection, e. g., a TLS connection. As mentioned above, it is not necessary that $\mathcal{T}$ is a computer system as k^1_{auth} could also written on a piece of paper by the administrator. But mostly it could be assumed that $\mathcal{T}$ is a specially secured and encrypted database. After sending the authentication key, $\mathcal{U}$ deletes (k^1_{auth}) securely.

$\mathcal{U}$ can now initialize the master log file by saving the first message `STARTING LOG FILE` in the log file as described next. For this step, k^1_{sign} is the actually used secret. Important is that the master log is normally generated only once per SLOPPI instance.

2) Saving New Log Entries: Let m be the log message of the log entry to be stored in the log file. As the master log file has only one entry per day, it can be assumed that there is enough time between saving the last entry and the actual one to generate a new authentication/signing key pair ($k^{n+1}_{\text{auth}}, k^{n+1}_{\text{sign}}$), while k^n_{sign} is the actual secret.

$\mathcal{U}$ now generates

$$m^* = (\textit{timestamp}, m, k^{n+1}_{\text{auth}})$$

and computes

$$e = \text{Enc}_{k^n_{\text{sign}}}(m^*).$$

The result e is the new log entry, which is written to the log file. Immediately after calculating the encrypted result, the keys k^n_{sign}, k^{n+1}_{auth}, which are not needed anymore, are erased securely from the system. Now the master log file only contains fully encrypted data and k^{n+1}_{sign} is the secret for the next log entry. The motivation for the data format used for m^* is the following:

- The *timestamp* is used to verify the time, at which a new event is logged in the master log. An abnormal high or low rate of entries in a specific time interval can indicate a system failure or an attack. To prevent any changes of the timestamp, this is also part of the encrypted data. item

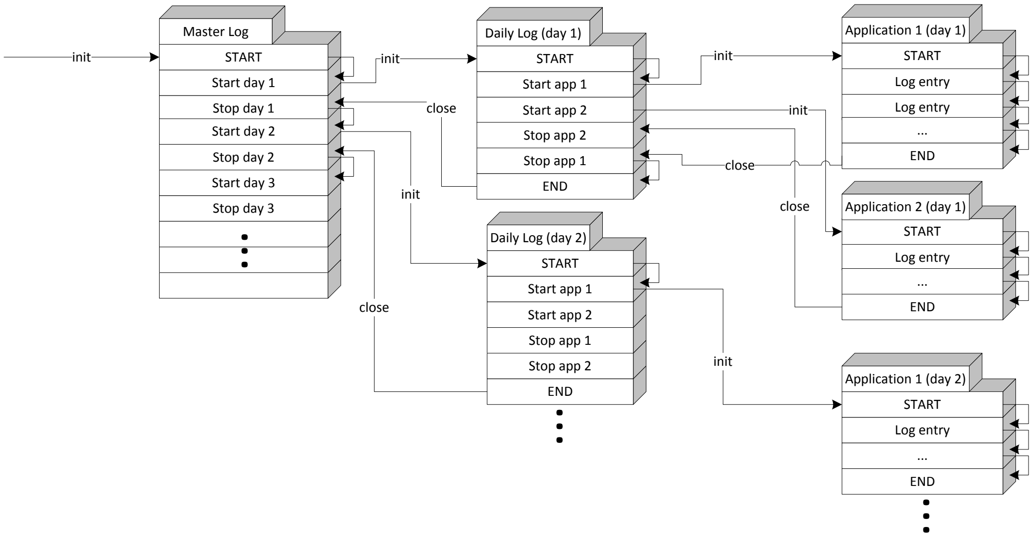


Fig. 4. Overview of all relevant log files.

The *log message* m has to be encrypted, to protect the main content of the log entry.

- k_{auth}^{n+1} is the *verification key* for the next log entry. This is the application of the forward integrity approach, because the necessary information to decrypt and to verify the next step are all available if the previous verification and decryption step is completed. As this is data very worthy of protection, it is naturally part of the encrypted data.

For example, let the first log message be

$$m = \texttt{STARTING LOG FILE}$$

and the actual secret k_{sign}^1. After generating $(k_{\text{auth}}^2, k_{\text{sign}}^2)$, $\mathcal{U}$ now composes

$$m^* = (1408096800, \texttt{STARTING LOG FILE}, k_{\text{auth}}^2)$$

and computes

$$e = \text{Enc}_{k_{\text{sign}}^1}(m^*)$$

$$= \text{Enc}_{k_{\text{sign}}^1}(1408096800, \texttt{STARTING LOG FILE}, k_{\text{auth}}^2),$$

which is written to the log file. Now, k_{sign}^1 and k_{auth}^2 are deleted securely.

3) Closing the Log File: During regular SLOPPI operation, there should not be the need to close the master log file. But in case of a system restart, a serious failure, or in the case where the storage requirement of the master log is too much increasing, there can be the desire to restart the master log.

If the master log file has to be closed gracefully, the last message `CLOSING LOG FILE` is saved into the log file. It is important that in this case it is not necessary to generate a new key pair and therefore, the next authentication key is also irrelevant. To fulfill the data format defined above, it is needed that the log entry consists the next authentication key, which is set to an empty string.

4) Content of Log Messages: As L_m is used as a meta log, which does not contain any application or system messages, the content of the log messages m are now specified. As mentioned before, the daily log is encrypted with a symmetric crypto scheme. Every day a new daily log is initialized by the system. The name and location of the created daily log is the variable p_1. Furthermore, the variable p_2 is the first entry in L_d and finally the variable p_3 appoints the necessary key for the log initialization step. m is then the concatenation of p_1, p_2, and p_3, e. g., `/var/log/2014-08-15.log;STARTING LOG FILE;VerySecretKey` together with $H(p_1, p_2, p_3)$.

Because of the need to detect manipulations of L_m, it is necessary that m also contains a hash value of p_1, p_2, and p_3. With the knowledge of $H(p_1, p_2, p_3)$ it is possible to detect where the decryption process failed exactly.

C. The SLOPPI Daily Log File

The main reason to use the daily log is to reduce the storage space requirements of the main log. It is quite unusual that the main log is initialized for a second time if the system is running normally. There are round about two entries per day, which have to be stored over a long time. The daily log could be deleted after all application logs mentioned in this specific daily log are deleted. Depending on the amount of running applications on a server it is not unusual that there is much more than one application log used on a system.

1) Log Initialization: Every day a new daily log has to be initialized if a daily log rotation is configured on the system. Otherwise, another initialization interval is used for starting a new daily log. At the beginning, $\mathcal{U}$ generates a symmetric key k_{sym} which is necessary for both, encryption and verification. This key is the initial secret and has to stored in a trusted space, e. g., on $\mathcal{T}$. As the SLOPPI approach does not need a separate $\mathcal{T}$, the already existing L_m is used as a trusted third party. As described above, it is unlikely that an attacker can get information out of L_m because this log file is fully encrypted. If the attacker has already compromised the system, no new log message can be trusted any longer and the attacker is able to modify any computation step. Therefore, it is secure to store the key inside of L_m. The key k_{sym} is now written together with the name and the path of the actual log file. This information is combined with the first message STARTING LOG FILE.

The actually used secret is now k_{sym}. $\mathcal{U}$ can then initialize the daily log file by saving the first message STARTING LOG FILE in the log file as described next.

2) Saving New Log Entries: To encrypt the important information of the daily log, a symmetric key scheme is used, e. g., AES (Advanced Encryption Standard). In difference to the master log, not all of the information stored in L_d can be encrypted, because parts are needed in plain text during later steps as detailed below.

Let (m, m') be the message that has to be stored in the log entry and $k_{\mathrm{sym}}^{\mathrm{old}}$ the actually used secret key. The precise meaning of (m, m') is defined below along with the log message content.

$\mathcal{U}$ now randomly chooses a new secret key $k_{\mathrm{sym}}^{\mathrm{new}}$. Because of the use of symmetric key schemes, this step is not computationally expensive. Analogous to the master log processing, an expended message is now generated by $\mathcal{U}$, which has the structure

$$m^* = (timestamp, m, k_{\mathrm{sym}}^{\mathrm{new}}, H(m', (timestamp, m, k_{\mathrm{sym}}^{\mathrm{new}})))$$

. With this, the log entry

$$e = (m', \mathrm{Enc}_{k_{\mathrm{sym}}^{\mathrm{old}}}(m^*))$$

can be computed, which is written to the log file. The hash value is stored for verification purposes, so it is possible to detect the exact log entry where a manipulation took place.

Immediately after calculating the encrypted result, the key $k_{\mathrm{sym}}^{\mathrm{old}}$, which is not needed anymore, is erased securely from the system. Now $k_{\mathrm{sym}}^{\mathrm{new}}$ is the secret for the next log entry. The motivation for the data format used for m^* is the following:

- The *timestamp* is used to verify the time when a new event is logged in the daily log. An abnormal high or low rate of entries in a specific time interval can indicate a system failure or an attack. To prevent any changes of the timestamp, this is also part of the encrypted data.
- The *log message* m has to be trivially encrypted, to protect the main content of the log entry. Besides m, there is also a part of information, which has to remain in plain text to use them in the typical usage of the SLOPPI tools.
- $k_{\mathrm{sym}}^{\mathrm{new}}$ is the *verification key* for the next log entry. This is the application of the forward integrity approach, because the necessary information to decrypt and to verify the next step are all available if the previous verification and decryption step is completed. As this is a data very worthy of protection, it again is naturally part of the encrypted data.

3) Closing the Log File: If the daily log file has to be closed, the last message CLOSING LOG FILE is saved. It is important that in this case it is not necessary to generate a new key pair and therefore, the next authentication key is also irrelevant. To fulfill the data format defined above, it is required that the log entry consists the next authentication key, which is set to an empty string. This message, the file name and location, the MAC of the entire log file, and the last generated key are committed to be stored in the master log.

4) Content of Log Messages: In the daily log, there are five types of messages. In difference to the master log, which contains only meta information, which is only interesting for verification purpose, the daily log also contains information, which is used in the daily use of the system. Therefore, these parts of the contained information of a message have to remain unencrypted. In the following, '_' means an empty string of zero bytes size.

- STARTING LOG FILE. This message only consists of the string, which should be encrypted.

$$(m, m') = (\texttt{STARTING LOG FILE}, '\_').$$

- CLOSING LOG FILE. This message only consists of the string, which should be encrypted.

$$(m, m') = (\texttt{CLOSING LOG FILE}, '\_').$$

- START APPLICATION LOG. This message contains the timestamp when the application log was initialized (this is in most cases the same timestamp, which is used above for saving the log file), the file name, and location of the application log. This information is saved in plain text inside the daily log because it is needed to identify the application logs, which are connected to the specific daily log.
  Furthermore, the message consists of the initialization key, the first message, and the file name and location of the application log in encrypted form. The encryption step happens when the log entry is being saved to the daily log. The initialization key of the application log is saved in the daily log because the daily log is the trusted third party $\mathcal{T}$ for the application log.
  This leads to $m =$ (START APPLICATION LOG, initialization key, first message, file name, location) and $m' =$ (START APPLICATION LOG, timestamp, file name, location).
- STOP APPLICATION LOG. Similar to the start message, this message contains in plain text a timestamp, the file name, and the location of the application log. The encrypted parts, which are also encrypted when saving the log entry, are the last key, the last message, and the file name as well as the location of the application log.
  This leads to $m =$ (STOP APPLICATION LOG, last key, last message, file name, location) and $m' =$ (STOP APPLICATION LOG, timestamp, file name, location).

- ROTATING APPLICATION LOG. In case of a log rotation procedure, it is necessary that the SLOPPI tool is able to know, which new log file was previously protected by the SLOPPI approach and which daily log this new log file is assigned to. For this reason, the message contains a timestamp and both file names, i. e., before and after a rotation, in plain text. As in the previous message type, there are – also for security reasons – the file names and locations as parts of the cipher text.
  This leads to $m =$ (ROTATING APPLICATION LOG, file name before, file name after) and $m' =$ (ROTATING APPLICATION LOG, timestamp, file name before, file name after).

It is important that all application logs, which have their starting message in a daily log, have to write their stopping message also to the same daily log. This is the reason why some contents in the log file are still in plain text. Otherwise, the logging engine would have to remember, which application log is connected to which daily log. This also means that it is possible that a daily log is still open when the next daily log is initialized. The ROTATING APPLICATION LOG message could be in later daily logs because the specifics of the log rotation algorithm are not known and it could be that a log rotation is performed only once a week.

D. The SLOPPI Application Log Files

In general, the application log files L_a can be protected by any approach presented in Section III and are not a mandatory part of the SLOPPI architecture. These log files can also be deleted for compliance reasons. It is also possible to use log rotation techniques to fulfill local data protection policies. It is necessary to mention that any information about an attacker, which is not detected during this period, will be lost and cannot be recovered. But this is not a drawback of the presented framework because this is necessary to fulfill the data protection legislation especially in Europe or in Germany, which mandates to erase any privacy protected data after seven days. In scenarios where the log files can only be read after a longer offline period, e. g., low power sensor-networks devices, the period to delete log files should be set individually so an administrator is able to analyze any log data before they are deleted.

SLOPPI's core components ensure that the essential requirements of secure logging are fulfilled: integrity and compliance. However, in high-security environments, additional characteristics are required, such as the confidentiality of application log entries with selective deletion of personal data. In this section, we present modular extensions to the SLOPPI architecture and workflows to provide such additional functionality. Due to the openness of SLOPPI's architecture, arbitrary other modules can also be implemented.

1) Application Log Message Encryption: In many cases application log files should be in plain text to allow other applications or human administrators to take a look into the log files. However, in other cases it is appropriate to protect the content of the log files from prying eyes.

In the basic SLOPPI approach, it is not required to encrypt the log entries in the application log files explicitly. But sometimes it becomes necessary that the application logs get encrypted. For this, SLOPPI provides an extension, which supports different encryption methods for the most common scenarios as discussed below.

In general, any existing encryption algorithm may be used for securing particular application logs. So, here the encryption extension for SLOPPI is specified in an abstract way assuming an encryption method using particular, respective secrets k_{sign} (encryption) and k_{auth} (decryption) to secure a particular application log L_a. For sure, $k_{\text{sign}} = k_{\text{auth}}$ if symmetric encryption scheme is used, which is generally recommended due to the amount of data that is typically logged to application logs.

Based on this assumption, there are still several different options how to actually encrypt the respective application log messages. These options must be bases upon the underlying scenario; the most common ones are discussed in the following:

- Sometimes different persons or groups are responsible for the administration of a system or service. As this sharing of responsibility mostly leads to more than one application logs, in which the different groups are divided, it is important to define, whether all application log files using the same key pair for encryption or different ones for each log file are chosen.
- Log messages in application logs can be very critical. For example, in a bank it is possible that each money transfer is also logged into the application log to prove that the transfer succeeded or failed. Assume that in this case it is insufficient if the log message is readable even for a short period of time. So here is the requirement that each log message is encrypted with a key, which is different from the previous message as well as from the following message.
- As SLOPPI should be runnable on nearly every system, it is sometimes difficult to generate a new randomly chosen key every time (especially in the case “one message, one key”). For this scenario, SLOPPI also supports an auto-derivation of the encryption key by using an appropriate key derivation function. This function can, for example, be based on Keccak or other algorithms that also support the generation of arbitrary-length key material.
- In general, log file analysis should be done nearly in real time to detect unusual events. For this it is not necessary to take a look into the log file after a predefined period of time, e. g., after the daily log rotation. If this is the case, it is sufficient to analyze the log files, while the verification process is running and so the log file encryption can start, for example, with a randomly chosen key pair. Otherwise, the verification and the log file analysis, i. e., the log file decryption, are different processes. In that case, the decryption key has to be deduced by applying an appropriate key derivation function. It is obvious that in this case the initial secret has to be stored on $\mathcal{T}$ otherwise,

decryption would also possible for any attacker.

The encryption extension for SLOPPI supports any combination of the described options above. Which options are chosen in a specific scenario depends on the characteristics of the application log file and environment used, e. g., the number of log lines per day, the speed of adding of log lines in maximum and in average, the computing power and memory resources of the system versus the confidentiality in the face of potential attackers for a day-period.

Even if the single key-pair option, i. e., for one or more log files one key per day, is chosen, a potential attacker will not be able to read any application log files of prior days, but he may be able to read the log file of the current day, as the single key pair is still accessible.

In any case, all necessary information about the choices made has to be stored initially in the respective start application log message in the daily log file. Therefore, the content of the startup application log message has to be extended in comparison to basic SLOPPI:

- $m = ($START APPLICATION LOG, initialization key, first message, file name, location) has to be changed to $m = ($START APPLICATION LOG, initialization key, encryption method, initial key or derivation info, iteration used?, chaining used?, first message, file name, location), where *encryption method* gives some information about the used encryption algorithm and also, which of the extensions above is used. *initial key or derivation info* is the place, where the used (initial) decryption key is stored. If this key is computable from previous information, the key information is replaced by information about the used key derivation function. The last two optional parts *iteration used* and *chaining used* define information about parts of the extensions described above.
- $m' = ($START APPLICATION LOG, timestamp, file name, location) does not require any changes.

All application log messages of the respective application log file have to be encrypted according to the choices made. Additionally, if the *chaining* mode was chosen, each newly selected decryption key for the next application log entry has to be concatenated with the proper message of the current application log entry before this concatenation is encrypted using the current encryption key. This leads to the following:

- $applogentry_i = E_i(decryptkey_{i+1}; msg_i)$ if chaining is used
- $applogentry_i = E_i(msg_i)$ otherwise

2) Application Log Message Enrichment: Log messages sent to SLOPPI can be parsed and enriched before being stored in an application log file. This brings the following benefits:

- The application-specific structure of log messages can be made explicit, for example, by marking data fields containing personal data or specific error codes. This simplifies processing SLOPPI application log files in log management tools or business intelligence software, such as Splunk.
- Recommendations on how the content of the log message should be visualized can be added.

Given log message m of length n, $c_1, \ldots, c_n$ denotes the individual characters that m is composed of. A *slice* is defined as tuple $\langle c_i, c_j \rangle$ with $i \leq j$ and denotes the boundaries of an arbitrary non-empty sub-string of m. Unique names are assigned to slices and can be used to describe their semantics. For example, if a network service's log message contains the IP address of a client, the resulting slice could be named *ipaddress*. The exact boundaries for each slice in any log message are determined by parsing rules. Typically, regular expressions will be used to parse a log message in order to determine the slice boundaries. If a parsing rule matches the given log message, the slice name and boundaries are appended to the log entry. For example, if the IPv4 address 10.31.33.7 is detected in a log message starting at c_i with $i = 27$, then the string `@ipaddress:27-36` is appended to the log entry.

In general, slices can be overlapping. For example, a whole log message may be slice-tagged as *service x's error message*, while a sub-string may be tagged *ipaddress*. However, for the personal data anonymization procedure discussed below, care must be taken that the two types of used slices do not overlap to ensure that one slice's hash value does not change when another slice is being modified. Otherwise, the a-posteriori anonymization of log files would break the verification procedure.

Slices are useful for two slightly different use cases: On the one hand, the SLOPPI tools can be instructed to verify and decrypt only selected slices; this allows for a fine-grained access model where system administrators can be restricted to which parts of single log file entries they are allowed to read – this allows for a more detailed access management than the traditional per-file or per-log-entry model found in most of today's implementations. On the other hand, slice names and ranges can be fed to other log entry processing tools, such as business intelligence software and log file visualization tools. This not only saves the overhead of parsing the same log message multiple times in different processing tools, but its true power lies in that the slicing can be integrity-checked. For example, it becomes obvious whether an IPv4 address has been recognized as such by looking at the slice names; compliance violations, such as not checking which parts of log messages contain personal data that must be removed after the maximum retention time become easy to spot for an auditor. Also, administrators do not need to wait until the maximum retention time has been reached to verify whether an anonymization batch run will modify the correct parts of the log messages.

E. Generalized Privacy Protection for the Application Logs

As introduced in the previous section, SLOPPI also provides a semantic tagging of log entries. This can be used to identify privacy-relevant data. With this knowledge it is possible to anonymize these relevant parts of the log file, while the remainder of the information can be stored untouched.

In general a log message is only a string without any semantics for the normal logging process. Because of that the normal logging process is not able to differentiate privacy relevant data. With SLOPPI and the application log message enrichment extension, it is possible to give the logging process the needed information to separate compliance sensitive data from other text. As the goal is to enable the a-posteriori anonymization of log files, the content of each log message is split into two categories:

- The part of the log, which has to be anonymized later on.
- Everything else.

To perform the seperation between anonymized and non-anonymized data, there has to defined a priori the exact log format. For example, in the SSH (Secure Shell) log files, the log entries are always in a uniformed format, e. g., on a standard Debian web server there is always at the beginning a timestamp, followed by the host name, the process name and number and finally the log message, which contains the authentication method, the user name and finally the source IP address and the port. In this example, only the username and the IP address is necessary to anonymize for fulfilling the legislation boundaries. With the help of regular expressions, the positions of this information can be found. In other examples, this might be more complicated, but in general each logging source has its own specified output format. If there are to many mixed entries in a log file, there is always the possibility to split this log file into more than one others.

As written above, for the protection of the application log any approach of Section III can be used. In the following the forward integrity approach is described; any other approach can be handled analogous.

For the integrity protection, a MAC of each log entries is used. This MAC would change if an anonymization is performed afterwards and a verification would be impossible. But with the knowledge of the relevant parts, it is possible to calculate two MACs. The first is the original MAC with all information, while the other MAC is calculated with already anonymized parts of the log message.

These two MACs are appended accordingly to the log entry. The verification process can now check both MAC values. If no anonymization has been performed yet, the original MAC can be used. Otherwise, the verification step uses the new MAC. For this it is important that the SLOPPI anonymization uses the same anonymization string every time. To ensure human readability, i. e., administrators should know, which parts of a log message have already been anonymized, a placeholder replacement string, such as XXX or $***$ should be used. For simple implementation, it needs to be of constant length; this also avoids issues regarding the de-anonymization success probability when using variable-length strings.

In general it is important to know, which log entry has been anonymized at what time. Because an attacker could otherwise, anonymize any log entry himself to cover up the traces. To prevent this, SLOPPI generates a new log entry in the daily log, which contains the message LOG ANONYMIZATION and also the information, about which log file and which log entries are being anonymized, e. g., all log entries between the timestamps 1408010400 and 1408096800. To avoid race conditions, this log entry is created after an anonymization batch run has successfully finished.

F. Generalized Privacy Protection for the Application Logs with Partial Encryption

Both extensions of sections IV-D1 and IV-E can be combined to allow for on the one hand the partial deletion of log entries, e. g., to support privacy for relevant data, and on the other hand for partial encryption of log entries, simultaneously. Assumed is an partial application log file L_a and its application log messages being partitioned into different slices (i. e., tagged parts):

- Different slices of application log messages are – either iteratively or in chained mode – to be hashed separately as described in Section IV-E; this is done using multiple initial hash secrets, which are being stored in the respective start application log message for the application log file in the daily log file.
- Particular slices of application log messages can in addition be iteratively encrypted similar to IV-D1, but only for the particular slice. Each slice is treated separately with different encryption methods and secrets. Therefore, the method identifier needs to be stored along with the initial description key for each slice in the respective application log message for the application log file in the daily log file.

In sum, this requires the slicing of log messages along with partial encryption of each slice and therefore adds some overhead to the application and daily log files.

V. SLOPPI Log Entry Verification Procedure

To verify log entries, the initial master key is needed. Each log entry in the master log is encrypted as $\mathrm{Enc}_{k^n_{\mathrm{sign}}}(m^*)$ with $m^* = (timestamp, m, k^{n+1}_{\mathrm{auth}})$. To decrypt the message only the authentication key is needed, which is stored during the log file initialization step. After the first log entry is decrypted, the authentication key to decrypt the second log entry is obtained implicitly, and so on. The first occurrance of a log entry, which cannot be decrypted gives proof that a manipulation of the log files, which has been caused by an attacker or a malicious administrator who has tried to blur his traces.

This verification step is to verify the master log and to obtain the verification keys for the daily log. As the entries in the daily log look like $\mathrm{Enc}_{k_{\mathrm{sym}}}(m^*)$ with the content $m^* = (timestamp, m, k^{\mathrm{new}}_{\mathrm{sym}})$, the first entry could be decrypted by using the symmetric key stored in the master log. The symmetric key for any other entries is in the message payload of the previous log entry. As above in the master log, it is not possible for an attacker to modify any log entry in such a way that the encryption step works correctly.

A. Master Log: Verification of Log Messages

To verify an existing master log, it is necessary to use the authentication key saved during the generation of the log file. With this key it is possible to decrypt the first entry, which leads to the next authentication key. This step can be performed until the actual last message or the literal `CLOSING LOG FILE` entry is reached. Because m consists of the necessary information about the daily log files, it is possible to verify any daily log that is still available. If a daily log has already been deleted, then this daily log and the connected application logs cannot be verified any more, but it is still possible to use the subsequent log entries of L_m. Regularly the master log file is not closed because it is intended that the master log runs endlessly. Therefore, the last timestamp has to be near the current timestamp, but the derivation depends on the daily log scenario.

In the case of a failure, e. g., in the storage device, it is possible that writing in the master log file is interrupted suddenly. In this case, the last message of the master log is not the `CLOSING LOG FILE` message and probably has a somewhat out-dated timestamp, depending on the daily log scenario. In this case, verification is only possible up until the last timestamp just as if the master log had been closed regularly and no new master log had been started.

B. Daily Log Verification of Log Messages

Because of the need to detect any manipulation of L_m, it is necessary that e also contains a hash value of the message. With the knowledge of this hash value it is possible to detect, where the decryption process failed. To verify an existing daily log, it is necessary to use the authentication key saved during the generation of the log file. With this key it is possible to decrypt the first entry, which in turn leads to the next authentication key. This step can be performed until the actual last message or the `CLOSING LOG FILE` entry is reached. Because m consists of the necessary information about the application log files, it is possible to verify any application log that is still available. If an application log has already been deleted, then this application log cannot be verified any more, but it is still possible to use the subsequent log entries of L_d.

In the case of a failure, e. g., in the storage device, it is possible that writing in the daily log file is interrupted suddenly. In this case, the last message of the daily log is not the expected `CLOSING LOG FILE` message; it may also have an out-dated timestamp. In this case, verification is only possible until the last timestamp just as the daily log has been closed regularly and no new daily log has been started.

Furthermore, the transfer of the data between the daily log and the master log can fail. In this case the master log doesn't get the information that a daily log was initialized or was closed. This leads to a point where the verification process fails because the last secret keys are already deleted and so it is impossible to recover them to verify the last log entry.

C. Application Log: Verification of Log Messages

The procedure for verifying application log entries heavily depends on the options chosen for encryption. Basically, the key material that is required for beginning with the verification is part of the daily log. If the application log is not encrypted, the verification is concerned only with checking the application log's integrity by calculating the hash values given the current application log file line by line, and comparing the result calculated during the verification process with the values stored in the log file. Any mismatch is an indicator that the system has been compromised after the timestamp of the previous entry, i. e., the one before the offending line.

If the application log is additionally encrypted, there are two verification methods. On the one hand, full verification requires decrypting the application log file line by line as the decryption key material required for the next line is either based on the current line's material (iterative mode) or stored encrypted in the current log entry (chaining mode). An application log file's full verification is therefore $O(n)$ with n being the total number of lines in an application log file. Full verification is successful when all hash values match and the last line decrypts to yield the end application log line. On the other hand, the partial, or basic, verification omits any decryption steps. In other words, it just ensures that the integrity-protecting hash values are correct. The advantage of this approach is that basic verification can be performed without supplying the decryption key. It therefore can be used with automatisms not trusted enough to be provided with the decryption password.

VI. SLOPPI Security Analysis

In the SLOPPI approach, the method authenticate-then-encrypt is used to secure for example the daily log, as the the message m is in a first step authenticated by a hash function H and then both (m and $H(m)$) are encrypted. As Krawczyk discovered, there are some problems by using authenticate-then-encrypt, which makes them vulnerable against chosen plaintext attacks [10]. But on the other side, by choosing the right encryption methods (e. g., CBC (Cipher Block Chaining) mode or stream ciphers) also the authenticate-then-encrypt method is secure. This security consideration has to be taken into account, if the SLOPPI framework is implemented.

Furthermore, it is possible to relinquish most of the calculated MACs by using authenticated encryption with associated data cipher modes [11]. By using these cipher modes, only the encryption step is necessary to perform to gain encryption and also authentication of the messages. On the other side it has to be evaluated, if the performance and memory needs of these ciphers are comparable to traditional cryptographic schemes.

On the one hand it is not possible for an attacker to gain information from the daily log or the master log as they are both encrypted with well-known crypto schemes. On the other hand it is not possible to delete any entry of the log files because during the verification step, the decryption of the entry would fail and the manipulation would become evident. Furthermore, assuming proper implementation, any used key is removed securely from memory immediately so no attacker could restore it.

The only possibility of an attacker is to be fast enough to gain access to the system and to shut the logging mechanism down before any log entry is written to the disk. This may eventually happen, e. g., when the attacker performs a DoS attack on the system before he is breaking in. However, implementations of the logging service of the system are able to prevent the system from this method of attack by aggregating multiple identical events before writing them to application log files; the same approach can be used for future SLOPPI implementations.

The confidentiality of log messages is related to the feasibility of unauthorized decryption. For SLOPPI, the following aspects need to be considered:

- The master log file is strongly encrypted in any of the operational modes described above, i. e., for the iterative mode as well as the chaining mode, which is applied to each individual application log file. The master log file uses public/private key encryption and therefore secure as long as either of both keys is kept out of an attacker's reach.
- The daily log file is also strongly encrypted in any of the operational modes. However, it uses a symmetric encryption scheme, which means its security relies on generating high-quality random key material, which an attacker must never see. Generating key material, e. g., by using pseudo-random number generators (PRNGs), typically works well on systems with good entropy pools, such as networked servers or interactively used machines. However, especially embedded systems without entropy-generating sensors may be bad at generating new key material. In this case, manually planting an unique random number seed that provides enough input for further key derivation during the system's estimated run-time before system deployment is mandatory.
- In the basic SLOPPI operating mode, only iteratively securely hashed key material is used. The initial hash secret is saved in the respective opening daily log message in the daily log. Therefore, the confidentiality of the daily log file, which has been discussed above, is sufficient to ensure the secrecy of this keying material.
- When SLOPPI is used to also encrypt log messages in the application log files, these log files make use of the same iterative or chained encryption:
  - The initial decryption key $k_{m,d,1}$ is being stored along with its initial hash secret in the respective daily log message about opening the application log; the confidentiality of the daily log protects this key material.
  - With each log entry $i, i \geq 1$, in the application log, the entry meta-data also contains the next iteration of the decryption key $k_{m,d,i+1}$.

When using SLOPPI with application log file encryption, using either iterative or chained key material generation deserves further discussion. The basis for both options is the generation of new key material in the beginning; it depends on the quality of the PRNG, similar to the daily log, which has been discussed above. Furthermore, key material that is no longer needed needs to be correctly and irrevocably be erased from memory, i. e., after the first log message iteration or after the application log file has been closed. The confidentiality also depends on the strength and quality of the used key derivation algorithm and its mode of operation, e. g., counter mode, feedback mode, or double-pipeline iteration mode. Similar to encryption, many options are available and SLOPPI does not prescribe, which one to use.

If SLOPPI is used without any iteration re-keying option, then an attacker may be able to read plain text messages that are already written to the application log file before the attacker gained control of the system, but limited to the particular application log file of the current day. For any other, already closed application log files, i. e., those from earlier days, the used key material should already have been erased irrevocably from memory.

If the auto-iteration SLOPPI mode is used, which is based on key-derivation functions to generate new key material during re-keying, and the proper removal of no longer needed key material from memory is ensured, then the attacker is unable to access the plain text of the written application log messages up until the current re-keying at the time of system compromise.

Finally, if the chaining SLOPPI mode is used, i. e., new key material is generated for each application log entry, the quality of the achieved protection again is based on the quality of the used PRNG and proper reliable removal of outdated key material from memory. Attackers cannot gain access to previously written log entries' plain text either. Depending on the cipher that is used, ideally both options only differ in the amount of new key material that needs to be generated.

As the application log files can be deleted independent of their encryption status – which is required to fulfill compliance policies regarding retention time – without violating their integrity status because the actual content of the application log message is irrelevant for an integrity check, the mode of encryption does not interfere with regular SLOPPI operations.

It must not be neglected that SLOPPI does not address the availability issue of log files. Attackers can remove traces of attacks by simply deleting log files. Although this leads to an alarming situation, i. e., administrators can be informed about missing, truncated, or integrity-violating files, SLOPPI does not guarantee full traceability of all logged messages. Other techniques, such as using write-only memory, would be needed as this problem is all but impossible to fix in software realistically, given today's operating systems and their potential vulnerabilities, i. e., even a write-once file system could be accessed in a reading fashion, e. g., after system reboot and re-parametrization by an attacker who gained administrative privileges.

VII. Conclusion

SLOPPI is a log file management framework that supports integrity-checking and confidentiality through encryption like other approaches, but has a special focus on compliance. Compliance with privacy and data protection acts can, at least in Europe, only be achieved by limiting the retention time of personal data, which may also be a part of log messages written to log files. SLOPPI therefore supports the fine-grained partial removal of log entries, while still ensuring the properly working integrity-monitoring of the other logged data. For example, German Internet service providers and any other providers of Internet-based services are obliged to remove personal data from log files after 7 days based on court verdicts that have become effective. Using the SLOPPI approach, log files older than the maximum personal data retention time can be modified to have all personal data removed, while still ensuring that the log file has not been tampered with otherwise.

In this article, we have presented the inner workings and security analysis of SLOPPI in more detail than our previous work [1] and specified two extensions to the original SLOPPI architecture. First, we introduced the concept of slicing log messages and use them for semantic tagging, which can also be used in conjunction with external tools, such as business intelligence applications, for a-posteriori removal of personal data from log entries, and applying encryption for fine-grained access management protecting read access to log file information. Second, we presented the two options of iterations and chaining to address re-keying for encryption application log files, which were not part of the original SLOPPI specification.

To this extent, SLOPPI has several important functional advantages over the previously state-of-the-art logging mechanisms. However, they come with the inherent overhead of additional cryptographic operations and, to a large extent, depend on the quality of random numbers generated for constructing key material, which can be a problem for low-interaction systems without sufficient entropy-generating pools. As future work on SLOPPI it is therefore planned to support adaptive protection mechanisms that vary the strength of the used key material depending on the sensitivity of the data it is applied to. For example, stronger cryptography could be applied to log message slices containing personal data, while trivial log entries are only protected using weaker mechanisms.

Acknowledgment

Parts of this work has been funded by the German Ministry of Education and Research (FKZ: 16BP12309). The authors wish to thank the members of the Munich Network Management (MNM) Team for helpful comments on previous versions of this article. The MNM-Team, directed by Prof. Dr. Dieter Kranzlmüller and Prof. Dr. Heinz-Gerd Hegering, is a group of researchers at Ludwig-Maximilians-Universität München, Technische Universität München, the Universität der Bundeswehr München, and the Leibniz Supercomputing Centre of the Bavarian Academy of Sciences and Humanities.

References

[1] F. von Eye, D. Schmitz, and W. Hommel, "SLOPPI – a framework for secure logging with privacy protection and integrity," in *ICIMP 2013, The Eighth International Conference on Internet Monitoring and Protection*, W. Dougherty and P. Dini, Eds. Roma, Italia: IARIA, Jun. 2013, pp. 14–19. [Online]. Available: http://www.thinkmind.org/download.php?articleid=icimp_2013_1_30_30021 [accessed: 2014-12-01]

[2] W. Hommel, S. Metzger, H. Reiser, and F. von Eye, "Log file management compliance and insider threat detection at higher education institutions," in *Proceedings of the EUNIS'12 congress*, Oct. 2012, pp. 33–42.

[3] M. Hoffmann, "The SASER-SIEGFRIED project website." [Online]. Available: http://www.celtic-initiative.org/Projects/Celtic-Plus-Projects/2011/SASER/SASER-b-Siegfried/saser-b-default.asp [accessed: 2014-12-01]

[4] S. Metzger, W. Hommel, and H. Reiser, "Migration gewachsener Umgebungen auf ein zentrales, datenschutzorientiertes Log-Management-System," in *Informatik 2011*. Springer, 2011, pp. 1–6. [Online]. Available: http://www.user.tu-berlin.de/komm/CD/paper/030322.pdf [accessed: 2014-12-01]

[5] M. Bellare and B. S. Yee, "Forward integrity for secure audit logs," Department of Computer Science and Engineering, University of California at San Diego, Tech. Rep., Nov. 1997.

[6] B. Schneier and J. Kelsey, "Cryptographic support for secure logs on untrusted machines," in *Proceedings of the 7th conference on USENIX Security Symposium*, vol. 7. Berkeley, CA, USA: USENIX Association, Jan. 1998, pp. 53–62.

[7] J. E. Holt, "Logcrypt: forward security and public verification for secure audit logs," in *ACSW Frontiers*, ser. CRPIT, R. Buyya, T. Ma, R. Safavi-Naini, C. Steketee, and W. Susilo, Eds., vol. 54. Australian Computer Society, Jan. 2006.

[8] D. Ma and G. Tsudik, "A new approach to secure logging," in *ACM Transactions on Storage*, vol. 5, no. 1. New York, NY, USA: ACM, Mar. 2009, pp. 2:1–2:21.

[9] A. A. Yavuz and P. Ning, "BAF: an efficient publicly verifiable secure audit logging scheme for distributed systems," in *ACSAC*, 2009, pp. 219–228.

[10] H. Krawczyk, "The order of encryption and authentication for protecting communications (or: How secure is ssl?)," in *Advances in Cryptology CRYPTO 2001*, ser. Lecture Notes in Computer Science, J. Kilian, Ed. Springer Berlin Heidelberg, 2001, vol. 2139, pp. 310–331.

[11] C. S. Jutla, "Encryption modes with almost free message integrity," in *Advances in Cryptology EUROCRYPT 2001*, ser. Lecture Notes in Computer Science, B. Pfitzmann, Ed. Springer Berlin Heidelberg, 2001, vol. 2045, pp. 529–544.

Design and Application of a Secure and Flexible Server-Based Mobile eID and e-Signature Solution

Christof Rath, Simon Roth, Manuel Schallar, and Thomas Zefferer
Institute for Applied Information Processing and Communications
Graz University of Technology
Graz, Austria
Email: {first name}.{last name}@iaik.tugraz.at

***Abstract*—Electronic identities (eID) and electronic signatures are basic concepts of various applications and services from security-critical domains including e-government, e-business, and e-commerce. During the past years, server-based approaches have been increasingly followed to implement these concepts. Unfortunately, existing server-based eID and electronic-signature solutions are usually tailored to a specific use case or deployment scenario. This renders a deployment of these solutions in arbitrary application scenarios difficult. To overcome this issue, we propose a flexible server-based eID and electronic-signature solution that can be easily deployed in arbitrary application scenarios while still providing a sufficient level of security and usability. The feasibility of the proposed solution is demonstrated by means of a concrete implementation. Furthermore, the claimed flexibility of the developed solution is shown by integrating it into a productive web-based time-tracking application. Its successful deployment and integration shows that the proposed solution provides a secure and flexible alternative to existing eID and electronic-signature solutions and that it has the potential to improve the security of security-critical services and applications from arbitrary domains.**

Keywords*—*e-government, e-business, eID, electronic identity, electronic signature, identity management, mobile security.

I. Introduction

With the rise of digital society, remote identification of users has become an increasing challenge as a growing number of services have been moved to the Internet. Design and development of concepts and solutions that provide remote identification of users have been a topic of interest for many years. We have recently contributed to this topic and have proposed and presented a server-based eID and electronic-signature solution that facilitates remote identification of users in arbitrary application scenarios [1]. In this article, we further delve into this topic and elaborate on our proposed solution.

In general, the need for reliable remote identification of users applies to public-sector applications (e-government) as well as to private-sector applications (e-commerce, e-business). Remote identification is usually achieved by means of a unique eID assigned to the user. An eID can for instance be a unique number, user name, or e-mail address. During authentication, the claimed identity (eID) is proven by the user. Reliance on secret passwords for authentication purposes is still the most popular and most frequently used authentication approach for online services. However, password-based authentication schemes have turned out to be insecure due to their vulnerability against phishing attacks and their poor usability, which often leads to the use of weak passwords that are easy to guess or easy to break [2][3].

Transactional online services from the e-government domain and related fields of application typically require reliable remote identification and authentication of users. Given the obvious drawbacks of password-based eID and authentication schemes in terms of security, two-factor authentication schemes have been developed for applications with high security requirements such as transactional e-government services. Current two-factor authentication schemes typically comprise the authentication factors *possession* and *knowledge*.

Popular examples of two-factor authentication schemes are smart card based solutions. During the authentication process, the user proves to be in *possession* of the eID token (i.e., the smart card) and proves *knowledge* of a secret PIN (personal identification number) that is specific to this eID token and that protects access to the token and to eID data stored on it. In most cases, smart cards additionally enable users to create electronic signatures (e-signatures). For this purpose, the smart card additionally stores a secret signing key and features hardware-based signature-creation capabilities. Access to the signing key and to the smart card's signature-creation functionality is again protected by means of two-factor authentication.

Smart cards are an ideal technological choice to combine the concepts of eID and e-signature, as they are capable to implement both eID and e-signature functionality. Thus, they are frequently used in security-critical fields of application such as e-business, e-banking, or e-government. For instance, various transactional e-government services that have been launched in Europe during the past years require users to authenticate themselves remotely with a personalized smart card and to complete online transactions by applying an electronic signature with the same card [4]. Unfortunately, smart card based solutions usually lack an appropriate level of usability, as they require users to obtain, install, and use an appropriate card-reading device in combination with the associated software [5].

Powered by the recent emergence of mobile communication technologies and motivated by the low user acceptance of smart card based eID and e-signature solutions, several *mobile* eID and e-signature solutions have been developed during the past years [6]. These solutions render the use of smart cards unnecessary, as they cover the authentication factor *possession* by means of the user's mobile phone. This way, mobile eID and e-signature solutions have the potential to significantly improve usability while maintaining a comparable level of security to smart card based solutions. This is supported by the fact, that, e.g., in Austria qualified signatures can be issued both, with smart cards and with mobile eID and e-Signature solutions.

Due to their improved usability compared to smart card based authentication schemes [5], mobile eID and e-signature solutions are in principle also suitable for use cases with lower security requirements. Unfortunately, existing mobile eID and e-signature solutions are usually tailored to the requirements of specific use cases and fields of application. This applies to most mobile eID and e-signature solutions that have been introduced and launched worldwide during the past years. Due to their limitation to specific use cases, these solutions can hardly be used in different fields of application. This leads to situations, in which most applications cannot benefit from the enhanced security and usability of existing mobile eID and e-signature solutions.

To overcome this problem, we propose a modular and flexible concept for mobile eID and e-signature solutions. The main idea behind the design of the proposed concept was to achieve a flexible solution and to maintain its compatibility to different use cases and application scenarios. Details of the proposed concept are presented in this article.

In Section II, we start with a brief survey of existing mobile eID and e-signature solutions and discuss their strengths and limitations. We then derive requirements of a mobile eID and e-signature solution that is applicable in arbitrary application scenarios in Section III. In Section IV, we introduce a technology-agnostic architecture for a mobile eID and e-signature solution that meets all predefined requirements. Based on the proposed architecture, we model three technology-agnostic processes that cover required functionality in Section V. The practical applicability and feasibility of the proposed solution is assessed in Section VI by means of a concrete implementation. The compatibility of this implementation with existing security-critical applications is evaluated in Section VII. Finally, conclusions are drawn in Section VIII.

II. Related Work

The reliable remote identification and authentication of users by means of two-factor based approaches has been a topic of interest for several years. For many years, smart cards have been the preferred technology to implement two-factor based authentication schemes. Thus, smart card based solutions have been introduced in several security-sensitive fields of application during the past decades. Especially in Europe, various countries, such as Austria [7], Estonia [8], Belgium [9], or Spain [10] have issued personalized smart cards to their citizens in order to reliably identify and authenticate them during transactional e-government procedures [4]. In most cases, smart cards do not only provide eID functionality but also enable users to create electronic signatures. This is of special importance in Europe, where electronic signatures can be legally equivalent to handwritten signatures according to the EU Directive 1999/93/EC [11]. The importance of electronic signatures is even strengthened by the EU Regulation on electronic identification and trusted services for electronic transactions in the internal market [12], which will soon replace EU Directive 1999/93/EC.

While smart cards work fine from a functional point of view, their usability is usually rather poor. This poor usability is mainly caused by the need for a card-reading device to physically connect the smart card to the user's computer. The need for additional drivers and software to communicate with the smart card and to integrate its functionality into security-critical applications also decreases the usability of smart-card technology in general and of smart card based eID and e-signature solutions in particular. This has for instance been shown by Zefferer et al. [5], who have set up a thinking-aloud test with 20 test users to determine and compare the usability of different approaches to provide eID and e-signature functionality. The conducted usability test has shown that users clearly prefer solutions that do not require smart cards and card-reading devices.

To overcome usability limitations of smart card based solutions, several mobile two-factor based eID and e-signature solutions have been developed during the past years. Surveys of mobile eID and e-signature solutions have for instance been provided by Ruiz-Martinez et al. [6] and Pisko [13]. All these solutions have in common that the factor *possession* is not covered by a smart card but by the user's mobile phone. All mobile eID and e-signature solutions that comply with demanding legal requirements, such as those defined by the EU Signature Directive, include some kind of secure hardware element, which is able to securely store eID data and to carry out cryptographic operations. Depending on the realization and location of this secure hardware element, mobile eID and e-signature solutions can be basically divided into the following two categories:

1) **SIM-based solutions:** Solutions belonging to this category make use of the mobile phone's SIM (subscriber identity module) to securely store eID data and to carry out cryptographic operations. In most cases, the use of a special SIM is required, as off-the-shelf SIMs do not feature the required cryptographic operations. Access to eID data stored on the SIM and to cryptographic functionality provided by the SIM is typically protected by a secret PIN that is only known to the legitimate user. This way, SIM-based solutions rely on two different authentication factors. This PIN covers the factor *knowledge* of the two-factor based authentication scheme. The factor *possession* is covered by the SIM itself, which is under physical control of the user. With regard to security, all SIM-based solutions share one conceptual drawback. As required cryptographic operations such as the creation of electronic signatures are carried out on the mobile end-user device, these operations and all the data that is processed by these operations are potentially prone to malware residing on this device. This is especially an issue on current popular smartphone platforms such as Android, which are known to be vulnerable against malware [14].
2) **Server-based solutions:** Server-based mobile eID and e-signature solutions implement the secure hardware element centrally, e.g., in a hardware security module (HSM) at the service provider. Such a solution has been proposed by Orthacker et al. [15]. The user's mobile phone does neither implement cryptographic functionality, nor store eID data. However, the mobile phone is an integral component of the

authentication process that is mandatory in order to gain access to centrally stored eID data and to carry out electronic signatures. Server-based solutions rely on two authentication factors. During signature-creation processes, the user needs to provide a secret password first. This password covers the authentication factor *knowledge*. Covering the authentication factor *possession* is more challenging. As the server-based secure hardware element is not under physical control of the user, this element cannot cover the authentication factor *possession*. This factor is again covered by the user's mobile phone, concretely by the user's SIM. To complete the authentication process, a one-time password is sent to the user's mobile device via SMS. This one-time password has to be returned by the user. This way, the user proves *possession* of the SIM, as the one-time password can only be received, if the user has control over the SIM. With regard to security, server-based approaches are conceptually advantageous, as they do not require critical data to present on potentially insecure and compromised mobile end-user devices. The weakest point of the server-based signature solution presented by Orthacker et al. [15] is probably the SMS-based user-authentication step, as SMS messages must not be assumed to be secure on certain smartphone platforms any longer [14].

For above-mentioned categories, concrete mobile eID and e-signature solutions have been developed and rolled-out on a large scale. For instance, SIM-based mobile eID and e-signature solutions have been set into productive operation in Estonia [16] and Norway [17]. A server-based mobile eID and e-signature solution has been in productive operation in Austria since 2009 [18]. Most existing solutions are tailored to a specific legal framework (e.g., national laws) or to a certain identity system (e.g., to a specific national eID system). For instance, the Austrian mobile eID and e-signature solution has been purpose-built for the Austrian official eID infrastructure and bases on data structures, protocols, and registers that are specific to the Austrian use case. The Austrian eID infrastructure has been discussed by Stranacher et al. [19] in more detail. Deploying this purpose-built solution in other countries would require major adaptations and cause additional costs. Similar limitations apply to most mobile eID and e-signature solutions that have been set into productive operation so far. Their purpose-built nature renders a use of these solutions in different fields of application difficult and expensive. This prevents a broad roll-out of mobile eID and e-signature solutions and prevents that all applications can benefit from their improved security and usability.

III. Requirements

The conducted survey on existing mobile eID and e-signature solutions has identified a lack of dynamically adaptable solutions that can easily be applied to arbitrary use cases. To tackle this issue, we propose a mobile eID and e-signature solution that can easily be used in arbitrary application scenarios. We have designed the proposed solution according to a set of requirements. These requirements have been extracted from an analysis of existing solutions and from published evaluations of these solutions such as the one presented in [5]. The derived requirements (R1-R5) are discussed in the following in more detail.

R1: **Flexibility regarding external components:** Mobile eID and e-signature solutions typically rely on external parties and components. Common examples for such components are certification authorities (CA), which bind a user's identity to her signing key, or identity databases (e.g., official person registers or company databases), which are required to derive eIDs for users. A generic mobile eID and e-signature solution must not be limited to certain external components but provide flexible means to integrate different external components (e.g., different CAs).

R2: **Avoidance of token roll-outs:** Long-term experience with smart card based solutions has shown that the roll-out of eID and e-signature tokens (e.g., smart cards, SIMs) causes additional (financial) effort and hence reduces user acceptance. Avoidance of necessary roll-outs of such tokens is hence a key requirement for usable mobile eID and e-signature solutions.

R3: **Usability:** The often disappointing user acceptance of smart card based solutions shows that usability is an important success factor of eID and e-signature solutions. For mobile eID and e-signature solutions, the following aspects need to be considered in particular in order to achieve an appropriate level of usability:

R3a: **Avoidance of installations:** Usable solutions must not require the user to obtain, install, and maintain additional hardware or software, as this causes additional effort.

R3b: **Platform and device independence:** Usable solutions must not be restricted to certain computing platforms, operating systems, or end-user devices, as users want to access services everywhere and at any time irrespective of their current execution environment.

R3c: **Location independence:** Usable mobile eID and e-signature solutions must not be bound to a certain mobile network but must also be accessible when roaming in foreign networks.

R4: **Security:** Security is an important requirement, as mobile eID and e-signature solutions are mainly applied in security-sensitive fields of application such as e-government or e-commerce. Hence, mobile solutions must assure a comparable level of security to other two-factor based eID and e-signature solutions and must be able to comply with given legal requirements such as the EU Signature Directive [11].

R5: **Easy and flexible deployment and operation:** From the service operator's point of view, mobile signature solutions should support an easy and flexible deployment as well as an efficient operation, in order to save installation, set-up, and operation costs.

Based on these requirements, we propose a generic and adaptable mobile eID and e-signature solution, which removes limitations of existing solutions. We introduce and discuss the concept of our solution in the next sections before providing details on its implementation in Section VI.

IV. Architecture

Mobile eID and e-signature solutions follow either a SIM-based or a server-based approach to store eID data and to create electronic signatures. Other approaches would be possible on smartphones but cannot be applied on standard mobile phones due to their limited capabilities. Considering the requirements defined in Section III, we have decided to follow a server-based approach for our solution. This means, that a central HSM is responsible for protecting all eID data as well as for computing electronic signatures. Since solutions based on server-side signatures have very limited hardware requirements on the user side, they are comparatively cheap, user-friendly, and flexible in their deployment, as no roll-out of tokens is required. This way, Requirement R2 and Requirement R5, which demand avoidance of token roll-outs and an easy and flexible deployment and operation, are fulfilled.

Furthermore, server-based approaches require no up-front investments in dedicated SIM cards and no requirements towards the mobile network operators (MNO), hence, the targeted user group is not limited to a single, or certain MNOs. This reduces barriers and enhances usability. Advantages of server-based signature-creation approaches in terms of usability and user acceptance have also been discussed by Zefferer et al. [5]. Thus, reliance on a server-based approach assures that Requirement R3, which demands a sufficient level of usability, is met.

A theoretic concept of a server-based mobile signature solution and an approach to store users private keys in a secure manner on a remote server have been proposed by Orthacker et al. [15]. The proposed solution fulfills the requirements of *qualified electronic signatures* as defined by EU Directive 1999/93/EC [11], which emphasizes the suitability of this concept for security-critical application scenarios. Furthermore, a server-based mobile eID and e-signature solution that is compliant to the EU Directive 1999/93/EC has been in productive operation in Austria for several years. This provides evidence that server-based solutions are capable to achieve a sufficient level of security and hence to meet Requirement R4.

On a high level view, our solution defines the three processes: *registration*, *activation* and *usage*. These processes have different properties regarding computational effort and security constraints. During registration, which is mainly a matter of legal and organizational requirements, the identity of the user is verified. Usually, it is sufficient to perform the registration only once per user. During activation, a new eID including a signing key and a certificate is created for a registered user. Activation is required once per life span of an eID. In the usage process, created eIDs and signing keys are used by the user for authentication purposes and to create electronic signatures. Details of the three processes will be provided in the following section.

The architecture of our mobile eID and e-signature solution reflects the three processes defined above. This is illustrated in Figure 1. The entire architecture is split into an inner part and an outer part. Components implementing functionality of the activation and the usage processes are executed within these two parts. As shown in Figure 1, each part has its own database to store required internal data.

This way, the architecture is mainly composed of two databases and the four core components *Activation Outer*, *Activation Inner*, *Usage Outer*, and *Usage Inner* as well as a central HSM as inner component. The split between inner and outer components is a security feature as it reduces the impact of a data loss in case a service connected to the outer world gets compromised. Communication between outer and inner components happens via a limited, pre-defined set of commands over an encrypted channel. The separation of the core components allows for a very flexible deployment where, e.g., the activation parts can run on different machines, a different network or, if the business process allows/demands it, without a remote access at all. Additionally, access rights can be granted more restrictively, as only the activation process requires write access to many fields in the databases. At the same time, it is also possible to deploy the complete service on a single machine, if this is the preferred deployment scenario. By defining separate components to cover the proposed solution's functionality, the chosen architecture meets Requirements R4 and Requirement R5, which demand a sufficient level of security as well as an easy and flexible deployment and operation.

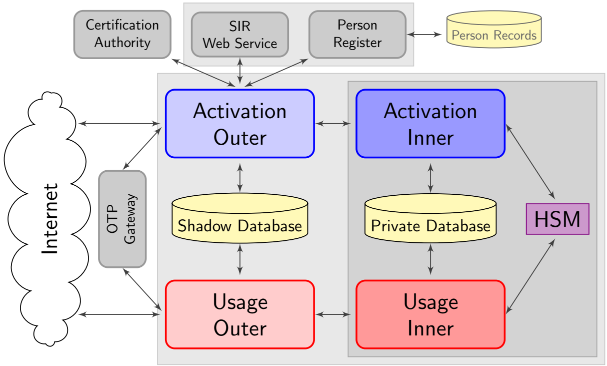


Figure 1: Overview of Core Components.

In addition to the four core components, the two databases and the HSM, the proposed architecture defines two internal and two external components. The external component *OTP Gateway*, which stands for one-time password gateway, is required during the registration, activation and usage process to send OTPs or activation codes to users. The internal component *SIR Web Service* is necessary for receiving so-called *Standard Identification Records* (SIRs). These records enable offline registrations, which will be discussed in detail in the next section. The components *Person Register* and *Certification Authority (CA)* are required during the activation process. While the CA is an external component, the Person Register is an internal component, which usually connects to an external database. The purpose of these components will also be discussed in detail in the next section. By clearly separating these components from the core components of the proposed solution, Requirement R1, which demands flexibility regarding external components, is already fulfilled on architectural level. The three processes, which build up our solution and cover its functionality, as well as all involved components, are described in the following section in detail.

V. Processes

The entire functionality of the proposed technology-agnostic mobile eID and e-signature solution is covered by the processes *registration*, *activation* and *usage*. The purpose of these processes is discussed in the following subsection in more detail.

A. *Registration Process*

During the registration process, data necessary to unambiguously identify a user is collected. Each user has to run the registration process at least once, before being able to use the proposed solution. To complete the registration process, the user has to prove her identity for example by means of a passport or an existing eID. In order to allow for a flexible setup of the registration process and to cover a broad range of legal and organizational requirements, the registration process has been designed to support different types of registration. These types of registration cover use cases from the e-government domain as well as use cases from related private-sector domains such as e-commerce or e-business. Furthermore, the proposed architecture is flexible enough to allow for an easy integration of further alternative registration types, in case they are required by the given use case. So far, the following four types of registration have been defined.

- **Registration via registration officer:** The identity of the user is verified face-to-face by a registration officer (RO) using official IDs, e.g., a passport or a driving license. After the verification of the user's identity, the RO manually registers the user in the proposed solution by filling the registration form with user-specific data.
- **Offline registration:** This registration type takes place in an asynchronous way. A user-data form has to be filled by an RO, after identifying the user similar to the registration type sketched above. After a validity check, the collected data has to be signed by the RO. The signed data is transmitted to the proposed solution and an activation code linked to the data is generated and passed to the user. The registration can be completed by the user at a later date using the issued activation code.
- **Self registration:** Self registration is carried out by the user herself with the help of an existing eID. While self-registration is common practice at online platforms, our solution relies on existing qualified eIDs for this purpose. The system verifies the user's identity by means of the provided eID and enables her to complete the registration afterwards on her own. An RO is not required for this type of registration, as the verification of the identity must have happened before during the activation of the existing eID.
- **Registration via trusted organization:** Many organizations have the legal requirement to identify their customers. Examples are bank institutes or universities. If a trust relationship with these organizations is established, existing identification data from these organizations can be used to register new users.

Figure 2 illustrates the general registration process of the proposed solution. The basic goal of the registration process is the creation of a Standard Identification Record (SIR) for a specific user. The SIR can be created using the four registration types sketched above. Irrespective of the applied registration type, a SIR is created which unambiguously identifies a user and provides this user basic access to the proposed solution.

Support of different types of registration allows for a very flexible setup of the registration process and covers a broad range of legal and organizational requirements regarding the registration process. This, in turn, contributes to a flexible operation of the proposed solution. This way, the proposed solution fulfills Requirement R5.

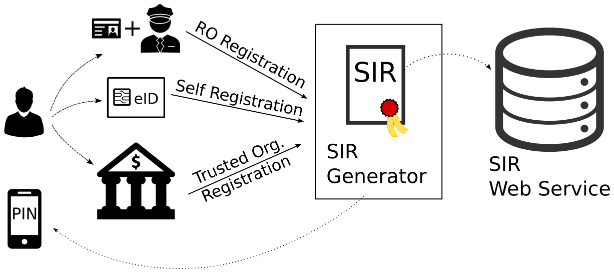


Figure 2: Registration.

B. Activation Process

After successful registration, users can run the activation process to create a new eID. For this purpose, the user needs to log-in to the proposed solution. This is only possible, if a valid SIR is available for the user (i.e., if the user has successfully completed the registration process) or if the user has already created an eID during an earlier activation process. In the former case, the user is unambiguously identified by means of the SIR. In the latter case, the user can log-in using the already created eID.

After a successful log-in, the user can create a new eID. For this purpose, the user is asked to fill the activation form. In general, the proposed solution supports multiple eIDs for each user. Therefore, the activation process can be run multiple times by each user. Each created eID can be managed separately. This enables users to have eIDs for different purposes, e.g., private and official affairs. During each activation process, a new cryptographic key pair is created for the user. This key pair can be used for subsequent signature-creation processes. Additionally, a certificate is issued to bind the user's identity to the created key. For each created eID, specific authentication data need to be defined by the user including a secret signature password (which will be verified against a regular expression pattern defined by the system administrator) and a mobile-phone number. The user has to prove possession of the specified mobile phone. This is achieved by means of OTPs that are sent to the user through an OTP Gateway.

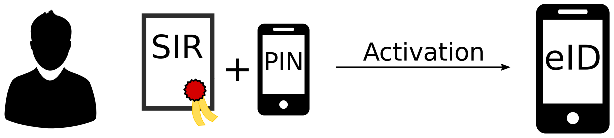


Figure 3: Activation.

In addition to the created key pair and the defined authentication data, also identity-related information, like full name and birth date, is assigned to the newly created eID. This information is obtained by the Person Register. The Person Register is a component that connects to an external database containing potential users of the service. Depending on the deployment and application scenario, this can be an existing official database like a central register of residence maintained by a public authority, an existing domain-specific database like the database of employees of a private-sector company, or a database specifically operated for this service that grows with every new registration. After fetching required identity-related information from the relevant database, the Person Register returns a signed data structure that contains the unique eID of the applicant and also the public key of the created signature key-pair. This way, it is possible to link a signature to a person for means of identification without the need to embed the unique eID directly in the signing certificate. By clearly separating eID functionality from e-signature functionality the users' privacy is assured. A similar concept has already been successfully applied in existing national eID solutions [20].

After completion of the activation process, a new eID has been created. This eID comprises signed identity-related information and a key pair (and certificate) for the creation of electronic signatures. Additionally, a secret password has been chosen and a mobile-phone number has been registered for the created eID, which are required during subsequent usage processes. The basic principle behind the activation process is illustrated in Figure 3.

C. Usage Process

After the successful completion of the activation process, the user can use the created eID to securely and conveniently authenticate at services and to create electronic signatures. To create an electronic signature or to authenticate, the user has to enter her phone number and signature password. If the data provided by the user can be verified, an OTP is sent to her mobile phone in order to verify possession of the mobile phone. If the user can prove possession of the mobile phone by entering the OTP, the requested signature creation is performed. The main concept behind the usage process is shown in Figure 4.

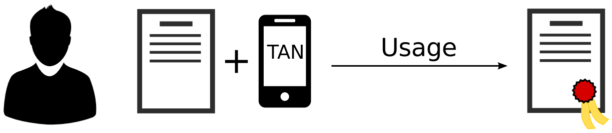


Figure 4: Usage.

VI. Implementation

Based on the proposed architecture and the defined processes, we have implemented a prototype to evaluate and demonstrate the applicability of our solution. This prototype has been named ServerBKU. The ServerBKU represents a server-based eID and e-signature solution. In the following subsections, we elaborate on the technologies used to realize the ServerBKU and discuss in detail the implementation of the ServerBKU's main processes.

A. Choice of Technologies

We have built our prototype implementation on a set of well-known and production-ready Java-based frameworks and libraries. This way, an efficient development process has been achieved and the probability of implementation errors has been minimized. Furthermore, reliance on appropriate frameworks assures that the prototype meets the requirements defined in Section III. Employed development frameworks, used libraries, and their underlying technologies are briefly introduced in this section.

The foundation of all implemented modules is the Spring Framework [21], which supports the development of modular and flexible software solutions. The basic underlying approach, followed by the Spring framework that enables a flexible design, is called dependency injection (DI). Following this approach, the dependencies of the various components are wired, i.e., injected, during runtime by the so-called inversion of control (IoC) container, a core component of the Spring framework. During development, concrete functionality, e.g., the OTP gateway, is abstracted by interfaces or base classes. Concrete implementations of the abstracted functionality are selected by configuring the IoC container. In the case of the OTP gateway, for instance, a special SMS-gateway implementation has been selected to implement the functionality of the OTP gateway interface. The flexible and easy selection of concrete implementations for abstract functionality enables a loose coupling of modules and allows software to be tailored to the specific needs of the use-case at hand. The loose coupling of modules also facilitates a test-driven development, as a single component can easily be tested without many dependencies. The concepts of DI and IoC are actually no unique features of the Spring framework. The Spring framework just implemented these concepts from the very beginning in 2002 in order to enable the development of flexible software.

Apart from the concepts DI and IoC, the Spring framework also provides templating mechanisms for various common tasks. This minimizes the amount of boilerplate code, which in turn reduces the chances of copy-and-paste errors and keeps the source code slim and readable. A prominent example of templates provided by the Spring framework is the Hibernate template. Hibernate [22] is an object-relational mapping (ORM) library, i.e., entries of relational databases are mapped to Java objects and vice versa. This way, most of the specifics of an underlying database can be abstracted by Hibernate. This enables an adoption of databases to the needs of certain deployment scenarios. By relying on the Spring framework, our prototype implementation, i.e., the ServerBKU, achieves a sufficient level of flexibility as demanded by the requirements defined in Section III.

In order to further improve the flexibility of the ServerBKU, a suitable proxy mechanism has been selected. This mechanism enables data exchange between different modules of the ServerBKU. For this purpose, the Java messaging service (JMS) API has been the technology of choice. Using this technology, the actual instance of an interface may run transparently on a different host. This way, it is possible to run the complete stack on a single machine or distribute the components over several servers. Apache ActiveMQ [23] has been chosen as implementation of the JMS API. Apache ActiveMQ supports out of the box redundancy and load balancing mechanisms. Furthermore, all exchanged messages can be protected via TLS secured channels. For this purpose, the IAIK iSaSiLk library has been used, which provides an extensible and highly configurable implementation of SSL 2.0 and 3.0 and TLS 1.0 and 1.1.

Libraries provided by IAIK [24] have also been employed to implement required cryptographic operations. Concretely, the IAIK provider for the Java Cryptography Extension (IAIK JCE) has been used to implement relevant functionality. Furthermore, the ServerBKU relies on the IAIK ECCelerate library to implement functions related to elliptic curve cryptography. To access the hardware security module (HSM), the ServerBKU uses the IAIK PKCS#11 Provider and Wrapper. The wrapper provides the Java Native Interface (JNI) to the hardware-dependent PKCS#11 library, while the PKCS#11 Provider implements a JCE provider for a specific hardware module. You can see an overview of the technologies used in Table I on page 10.

Besides appropriate development frameworks and cryptographic libraries, also a suitable technology to implement the required OTP gateway has finally been selected. Concretely, the ServerBKU has been defined to use transaction numbers (TAN), which are generated randomly, delivered via an SMS gateway to cover the functionality of OTPs and the OTP gateway. The employed SMS gateway operator provides an proprietary interface that enables the delivery of SMS messages via HTTP POST.

To assure the security of the ServerBKU, appropriate technologies have been chosen to assess our implementation by means of systematic security analyses. To follow an approved approach, the ServerBKU has been evaluated regarding the most recent critical risks according to OWASP [25]. Risks and flaws proposed by OWASP to be investigated are for example different types of code injection, Cross Site Scripting (XSS) or Cross Site Request Forgery (CSRF) amongst others. Analyses have been carried out using a white-box testing approach, as this method reveals most implementation errors. Several tools exist that facilitate such tests. Examples are Burp Suite [26] and several useful browser plugins that for instance allow the editing of cookies. Following the white-box approach allows the auditor having knowledge of the internal structure of the project, like the knowledge of libraries and frameworks in use, as well as having access to the source code. This way, the ServerBKU has been systematically and reliably assessed in terms of security.

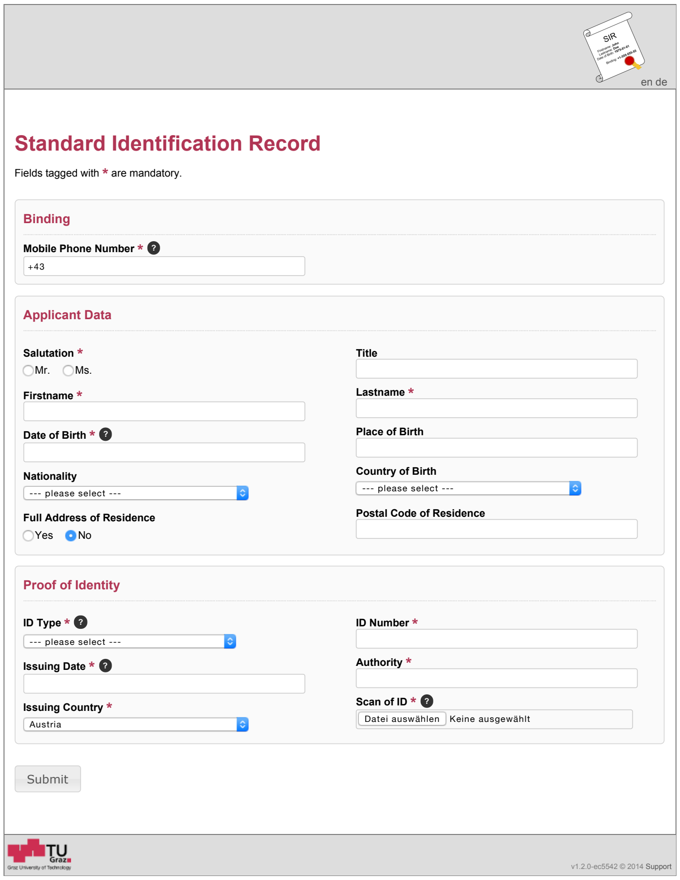
B. Realization of Processes

Based on selected technologies, development frameworks, and libraries, the three processes defined in Section V have been implemented. The implementation of these processes is described in detail in the following subsections.

1) Registration Process: In this step, the user has to prove her identity. Our implementation supports all types of registration defined in Section V. In a traditional setup, registration happens at the office of the RO. For this scenario, our implementation provides a web-based UI, through which the RO can register the user in the system by entering user data after identity verification. This UI is shown in Figure 5.

However, in some situations it might be beneficial for the RO to travel from user to user. This requires means to carry out asynchronous offline registration, as access to the ServerBKU is potentially not available at the user's place. To support this type of registration, the ServerBKU supports registration of users via SIRs created offline. A SIR contains information to identify a person, information about the ID used to verify the identity of the user, a binding towards a hardware token, i.e., a mobile phone for the use case at hand, and the electronic signature of a RO. Alternatively, a SIR can also be signed by a trusted partner, e.g., a bank or a university. This corresponds to the fourth type of registration listed in Section V. SIRs can be created from data entered by the RO (or trusted partner) or by the user using additional software. During the creation of the SIR, an activation code is generated and delivered to the user, cf. Figure 2.

Created SIRs must be sent to the ServerBKU's external SIR web service component via SOAP. The SIR webservice verifies the validity of a provided SIR by means of its electronic signature. If this verification succeeds, the SIR is stored in the user database of the ServerBKU. The user can use the stored SIR together with the activation code at a later date to start the activation process. The ServerBKU supports different front-ends that enable this type of registration. Initially, we developed a simple, yet comprehensive, stand-alone application based on Spring MVC. This application can be used on mobile devices in case of traveling ROs and supports the RO in creating SIRs (Figure 5). Furthermore, we developed an interface component that enables a traveling RO to take a picture of the ID of the user to be registered. The required data is extracted from this picture using optical character recognition (OCR). From these data, the required SIR is finally created. By supporting differnet means to create SIRs offline, the ServerBKU facilitates the offline registration of users.



Standard Identification Record

Fields tagged with * are mandatory.

Binding

Mobile Phone Number * ?
+43

Applicant Data

Salutation *
○ Mr. ○ Ms.

Title

Firstname *

Lastname *

Date of Birth * ?

Place of Birth

Nationality
--- please select ---

Country of Birth
--- please select ---

Full Address of Residence
○ Yes ● No

Postal Code of Residence

Proof of Identity

ID Type * ?
--- please select ---

ID Number *

Issuing Date * ?

Authority *

Issuing Country *
Austria

Scan of ID * ?
Datei auswählen Keine ausgewählt

Submit

Figure 5: Offline Registration.

To cover the last registration type, the ServerBKU provides a UI for the user. This UI is similar to the one developed for registrations vio ROs. It allows the user to carry out a self-registration in case she has already a trusted eID, e.g., smart card. As the user is identified and authenticated by means of this eID, no RO is necessary to complete the registration process.

2) Activation Process: In this process, the user creates and activates a new mobile eID. The activation process offers again a web-based interface. It has been developed using Java Server Faces (JSF) 2.1 [27] and Primefaces [28] for the frontend. The decision to use a different technology to create the UI is based on the rich set of UI components that is part of Primefaces. This facilitates development of a flexible, easy to use, role/permission-based interface in a short amount of time.

If the registration was performed in the classical way including an RO or as self-registration, the activation process starts automatically after registration. A pre-registered user can start the activation any time and independent of the registration process by submitting the received activation code and her telephone number to a specific URL. This way, available user data is automatically pre-populated as far as possible in the provided activation form by extracting the corresponding data from the SIR received from the database. Additionally, required data such as a signature password and a revocation password have to be entered by the user into the activation form.

As users may activate an arbitrary number of mobile eIDs for each phone number, activated eIDs have to be distinguishable by the system. This is achieved by the SHA-512 hash of the phone number and the signature password that have been selected by the user. Consequently, passwords have to be unique per telephone number. As the phone number is usually constant, a unique password has to be chosen for each eID. To verify the user's phone number and the possession of the device, a random OTP is generated, sent to the user's phone, and queried at the web interface. If the user enters a wrong OTP too often or if the code has expired, the activation process is aborted. The length and appearance (e.g., numeric, alphanumeric, etc) of this OTP as well as the number of trials and the time of validity is configurable. The user has also the possibility to resend the OTP a configured number of times in case the message gets lost on its way.

After the user has proven possession of the mobile phone, a signing key-pair for the user is created in the HSM. The private key is then wrapped by and exported from the HSM and securely, i.e., encrypted, stored in the ServerBKU's database. For details on the encryption scheme see below. Additionally, a certificate signing request (CSR) is generated. The public key is extracted from the CSR and sent to the Person Register together with additional data such as name and date of birth, which are required to identify the user in the Person Register. The Person Register returns a signed data structure that contains the unique eID of the user and the public key of the created signature key-pair. The returned signed data structure is, again encrypted, stored in the ServerBKU's database. Subsequently, an end-user certificate is requested from the CA using the already created CSR. The obtained certificate is stored together with the private key and the created eID data in the database.

The encryption of stored user data is based on a secret signature password, which the applicant chooses during the activation process. The ServerBKU relies on a hybrid encryption scheme as suggested by Orthacker et al. [15]. Here, the user has an additional encryption key-pair ($K_{\text{enc}}^{\text{pub}}$ and $K_{\text{enc}}^{\text{priv}}$), which is generated alongside the signing key-pair. The private key is then encrypted ($EK_{\text{enc}}^{\text{priv}}$) under the users signature password (PW_{sig}) and stored in the database. This happens only once during the activation phase.

$$SK_{\text{PW}} = \text{derive}(PW_{\text{sig}}) \tag{1a}$$

$$EK_{\text{enc}}^{\text{priv}} = \text{encrypt}(K_{\text{enc}}^{\text{priv}}, SK_{\text{PW}}) \tag{1b}$$

To encrypt a plain message M, a random symmetric key SK_{rand} is generated. This random secret key has to be encrypted for the user using her public encryption key (2b) and stored together with the cipher text (2a). However, this does not involve data from (1a) and (1b).

$$EM = \text{encrypt}(M, SK_{\text{rand}}) \tag{2a}$$

$$ESK = \text{encrypt}(SK_{\text{rand}}, K_{\text{enc}}^{\text{pub}}) \tag{2b}$$

This enables encryptions of data on behalf of the user without knowledge of the user's signature password. The decryption, however, requires the consent of the user, which she gives by providing the signature password (3a).

$$SK_{\text{PW}} = \text{derive}(PW_{\text{sig}}) \tag{3a}$$

$$K_{\text{enc}}^{\text{priv}} = \text{decrypt}(EK_{\text{enc}}^{\text{priv}}, SK_{\text{PW}}) \tag{3b}$$

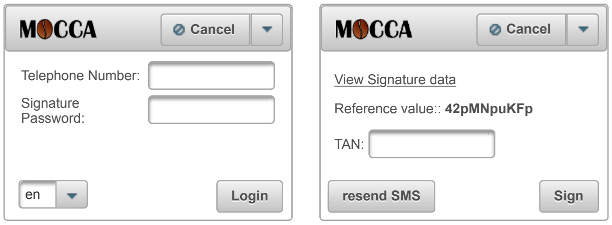
$$SK_{\text{rand}} = \text{decrypt}(ESK, K_{\text{enc}}^{\text{priv}}) \tag{3c}$$

$$M = \text{decrypt}(EM, SK_{\text{rand}}) \tag{3d}$$

After the generated certificate has been stored in the Server BKU's database, the user gets a notification per e-mail that the activation of the eID was successful. This finally completes the activation process.

Apart from the actual activation process, the implemented user interfaces also provide additional functionality. For instance, an interface has been implemented for ROs to perform activations on behalf of someone else as a usability feature. Furthermore, an interfaces is provided for each user that facilitates the management of eIDs, both for the user and also for a support team. This interface is shown in Figure 7. Finally, an administration UI has been developed that allows the definition and assignment of roles.

3) Usage Process: The usage process has been developed alongside the activation process and therefore is built on the same technologies, i.e., JSF [27] and Primefaces [28]. The interfaces are reduced to the bare minimum required for authenticating users and authorizing the creation of electronic signatures. This facilitates an easy integration of the Server-BKU into arbitrary third-party applications. The two forms that are used during the user-authentication process are shown in Figure 6.

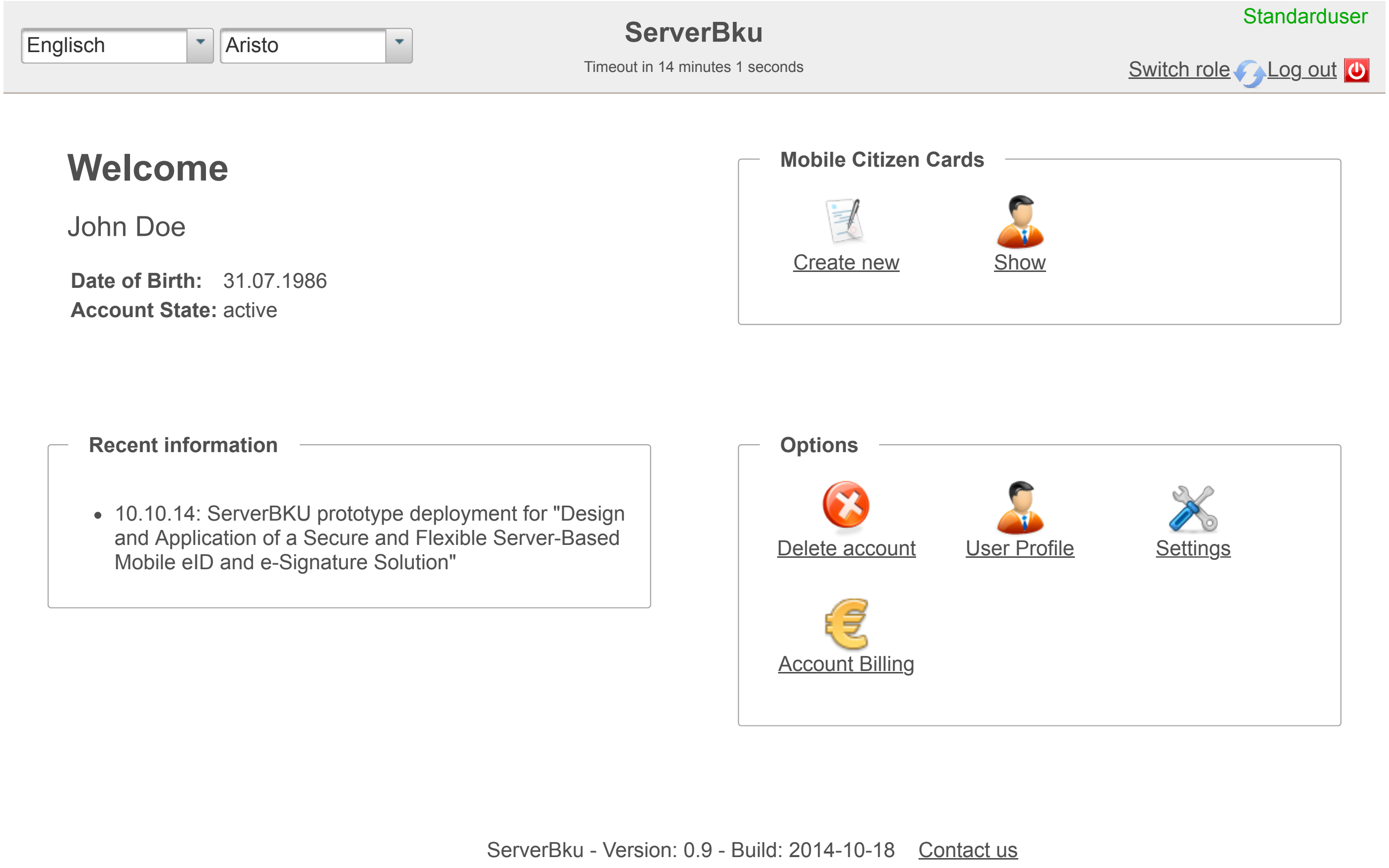


Reference value:: 42pMNpuKFp

(a) Login (b) TAN Verification

Figure 6: Interface of the Usage Process.

The signature-creation process starts with the receipt of an appropriate HTTP POST request at the web interface provided by the ServerBKU. The system returns the form shown in Figure 6(a), where the signer has to provide her phone number and signature password. The signature password is used to decrypt a private key that is part of the hybrid encryption used to securely store user-related data. Thus, neither the decryption



Englisch | Aristo

ServerBku

Timeout in 14 minutes 1 seconds

Standarduser

Welcome

John Doe

Date of Birth: 31.07.1986
Account State: active

Recent information

- 10.10.14: ServerBKU prototype deployment for "Design and Application of a Secure and Flexible Server-Based Mobile eID and e-Signature Solution"

ServerBku - Version: 0.9 - Build: 2014-10-18

Figure 7: Activation Management.

of the user's signature key has to take place before the two-factor authentication is complete, nor must the signature password be stored in a session. In order to prevent brute force attacks, the account for a given phone number gets locked a configurable period of time if too many unsuccessful log-in attempts are recognized.

If the user authentication was successful, two random values are generated: the OTP, i.e., the TAN, and a reference value, which is displayed in the TAN verification form (Figure 6(b)) and in the SMS that is used to deliver the TAN. This way, a link is provided between the TAN and the current session. Next, the service sends the TAN via the OTP gateway to the user's mobile phone in order to verify its possession.

After verifying the reference value received by SMS against the reference value displayed in the TAN verification form, the user enters the received TAN into the form. The form also provides a link to display the signature data. This enables the user to check the data to be signed prior to authorizing the signature creation. If the user has been successfully authenticated, the user data is read from the database and decrypted using the user's private key of the hybrid encryption scheme. Then, the still-wrapped private key of the signing key-pair is loaded into the HSM where it is unwrapped. Thus, the users private signing key is never accessible in a usable form outside the HSM. Finally, the unwrapped key is used inside the HSM to create an electronic signature on behalf of the user. After successful completion of the signature-creation process, the unwrapped key is discarded and the created signature is returned to the requesting entity.

VII. Evaluation

The ServerBKU shows that the server-based signature solution proposed in this article can be implemented in practice. To further evaluate its applicability in real-world use cases, we have deployed the ServerBKU in-house and linked it to an already existing application. We elaborate on this deployment and on the evaluation of the ServerBKU in this section. For this purpose, we first introduce the in-house application Timesheep, which has been used to evaluate the ServerBKU. We then show how the ServerBKU has been integrated into Timesheep to evaluate its applicability.

A. *Timesheep*

In the past, our organization used simple Excel sheets to record efforts, i.e., working hours, for employees and projects. Each employee had to fill out an Excel sheet with the efforts he spent on assigned projects. These Excel sheets were printed and had to be signed by the user by hand. After signing, the Excel sheets were forwarded to the group leaders. In the last step, responsible group leaders had to sign the Excel sheets themselves, in order to approve them. Signed Excel sheets were archived for project calculations. This process was cumbersome for several reasons including the following ones.

- Employees often forgot to fill out their Excel sheets in time and had to be reminded frequently.

- Employees sometimes forgot to print and sign Excel sheets, which caused delays in project calculations.
- Excel sheets had to be maintained and forwarded to group leaders manually, representing a potential source of error.
- When out of office, group leaders were not able to sign Excel sheets resulting in delays in project calculations.

To overcome these problems, our organization now uses a web-based time tracking tool called *Timesheep*. It tracks the efforts, i.e., working hours, for each employee and project. Timesheep runs on a virtual machine, which is only accessible from our internal network or through a Virtual Private Network (VPN) connection. Timesheep runs on similar technologies like the ServerBKU. It uses the Spring Framework [21] as a basis for modular and flexible development. Hibernate [22] is used to make the implementation independent from the underlying database. In addition, Spring Roo [29] has been used for fast prototyping. Spring Roo has been configured to use Spring Web MVC [30] as the web-rendering framework. To extend Spring Web MVC's tagx components, Prime UI [31], handsontable [32] and vis.js [33] have been used. You can see an overview of the technologies used in Table I.

TABLE I: Choices of Technologies Overview

Technologies / Application	ServerBKU	timesheep
Java	X	X
Spring Framework	X	X
Spring Roo		X
Spring WEB MVC		X
handsontable		X
vis.js		X
Primefaces	X	
Prime UI		X
Hibernate	X	X
Apache ActiveMQ	X	
IAIK iSaSiLk	X	
IAIK JCE	X	X
IAIK Eccelerate	X	
IAIK PKCS#11 Provider and Wrapper	X	

Timesheep defines several roles within our organization, in order to model required functionality:

- **User:** The first role is the role of normal users, i.e., employees. Users must have an easy access to Timesheep to track their efforts. This is achieved by providing a simple web-based interface, which can be accessed by using any common web browser.
- **Group Leader:** The second role is the role of group leaders. Group leaders must be able to access tracked efforts of employees assigned to their group, check these efforts, and approve, i.e., sign, them. Furthermore, the group leaders must be able to plan projects based on the budget of a project and on the efforts assigned users can raise until the project deadline.
- **Administrator:** The third role is the role of the administrator. Timesheep was developed to minimize the efforts administrators have to do. New employees are automatically added to the system with the necessary information for Timesheep to work. This is achieved by linking Timesheep to our in-house domain log-in system. This approach has also been followed by the ServerBKU, in order to avoid double registrations and to achieve maximum comfort. Thus, employees were able to log-in to Timesheep and to the ServerBKU with their domain log-in name without an additional registration process.
- **Financial Department:** The fourth role is the role of the financial department. Every project has milestones and a defined end date. At every milestone and end date, the current costs of the project must be calculated and submitted. Every submission is double checked by external financial auditors. Therefore, the auditors need the efforts done by each user on each project. After the project calculations are accepted by the auditors, tracked efforts must be protected against subsequent changes.

Based on these roles, Timesheep provides all required functionality to facilitate the tracking of efforts and the generation of time sheets. As all data are collected by one central web application and stored in one central database, required processes can be automated and delays caused by the manual tracking of efforts and processing of Excel sheets can be eliminated.

Although Timesheep has significantly improved the tracking of efforts in our organization, room for improvement could still be identified. The main drawback of Timesheep was its reliance on handwritten signatures. Even though efforts are tracked in electronic form and stored centrally in a database, employees and group leaders still had to sign generated time sheets per hand. Even though means to sign documents electronically exist, these means were not integrated into Timesheep.

To overcome this issue, we have integrated the ServerBKU into Timesheep. This way, Timesheep has been enhanced by means to electronically sign generated time sheets. Furthermore, this integration evaluates the practical applicability of the ServerBKU. The integration of the ServerBKU into Timesheep is discussed in the following section.

B. Combining Timesheep and ServerBKU

After all required efforts have been entered by the user and approved by the group leader, Timesheep creates a PDF-based time sheet called *monthly timesheet*. The monthly timesheet has to be signed by both the user and the responsible group leader. By integrating the ServerBKU, this signing process has been improved in terms of efficiency. For this purpose, we have integrated the ServerBKU's signature process seamlessly into Timesheep in order to achieve the best usability possible.

Timesheep is organized into multiple modules as shown in Figure 8. Each module fulfills a specific purpose for our organization.

- **Time Tracking Module:** This module offers a web-based interface for users to store their efforts.
- **Planning Module:** This module provides group leaders project planning and budget estimation based on user's contracts and employment status, and on project deadlines. Calculated values, e.g., hours, budgets, etc., are the so called *target* values.
- **Monthly Timesheet Module:** This module is responsible for generating and managing monthly timesheets. This module is also responsible for handling the signature process with the ServerBKU.
- **Financial Module:** This module offers our financial department the possibility to check how much a project did actually cost. These values are the so called *actual* values and may differ from the *target* values

defined by the Planning Module. Actual values will be submitted to external financial auditors.

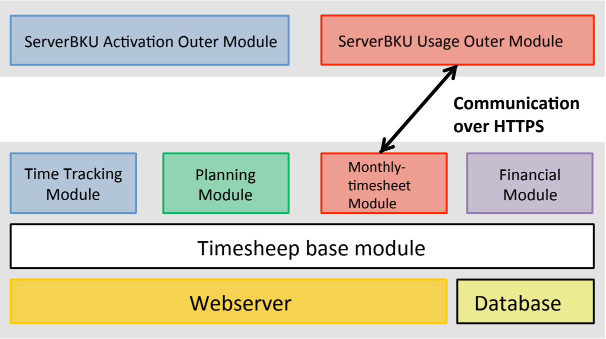


Figure 8: Timesheep Modules.

Figure 8 shows that the monthly timesheet module is responsible for the generation and management of timesheets. Hence, this module needs to be enhanced, in order to integrate the ServerBKU into the timesheet management and signing process. The ServerBKU-based signature process of a timesheet is shown in Figure 9. This figure shows how the monthly timesheet module interacts with components of the ServerBKU to electronically sign generated time sheets.

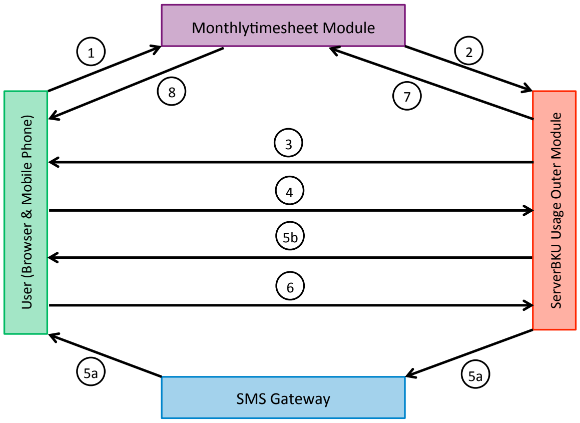


Figure 9: Work Flow of the Signature Process.

In the initial Step (1), the user starts the signature-creation process by clicking a button in his browser. A new browser window opens, in which all further communication between the ServerBKU and the user takes place. In Step (2), the monthly timesheet to be signed is sent to the ServerBKU. In Step (3), the ServerBKU prepares a PDF Advanced Electronic Signature (PAdES) and displays an authentication form to the user as shown in Figure 6(a). After that step, the user enters her phone number and signature password (Step (4)). In the next step, the ServerBKU sends a generated TAN to the user's mobile phone (Step (5a)) and displays the TAN verification form (Step (5b)). Next (Step (6)), the user enters the received TAN in the TAN verification form as shown in Figure 6(b). If the TAN provided by the user was successfully verified, the ServerBKU signs the monthly timesheet and sends the signature back to Timesheep (Step (7)). The browser window opened in Step (1) closes. Timesheep receives the signature, verifies it and notifies the user that the signature has been successfully verified. This is covered by Step (8). Signed monthly timesheets are stored in Timesheep's database and can be provided to external financial auditors to report efforts done by users.

By integrating the ServerBKU into our in-house application Timesheep, two goals have been reached. First, the process of signing time sheets containing tracked efforts of employees has been improved in terms of efficiency and usability. Even though the performance of the developed solution has not been systematically measured so far, related work on the usability of server-based signature solutions indicate that these solutions are advantageous in terms of usability [5]. This is also supported by first practical experiences gained with the ServerBKU-enhanced Timesheep instance. These experiences show that the integration of the ServerBKU improves the user acceptance of Timesheep and helps to reduce delays in reporting efforts to the financial department. A systematic measurement of the concrete usability and performance improvement that has been reached by integrating the ServerBKU into Timesheep is regarded as future work. Second, integration of the ServerBKU into our in-house application Timesheep shows that the proposed server-based signature solution in general and its concrete implementation ServerBKU in particular are applicable in practice and can be smoothly integrated into existing applications. Thus, the proposed server-based signature solution and the ServerBKU have been evaluated successfully.

VIII. Conclusion

In this article, we have proposed, presented, and discussed an enhanced server-based eID and e-signature solution. Based on a set of relevant requirements, we have developed an appropriate architecture for the proposed solution first. We have then carried this architecture over to a concrete implementation called ServerBKU using common state-of-the-art technologies. A test deployment of this implementation is publicly available online and can be accessed for test purposes [34]. Furthermore, the practical applicability of the ServerBKU has been evaluated by integrating it into the time-tracking tool Timesheep.

Even though the ServerBKU is ready for productive use, there are still some open issues that are regarded as future work. First, we need to gain more practical experience with our solution especially with regard to different deployment and application scenarios. Although first empirical results obtained by integrating the ServerBKU into the time-tracking tool Timesheep are promising, further experiences are required to further develop and optimize our solution. Second, we want to systematically measure the efficiency of the ServerBKU, in order to identify potential usability limitations. For instance, Single Sign-on solutions could help to reduce required user interactions and, hence, improve efficiency and usability.

While the concept of server-based eID and e-signature solutions is not completely new, the ServerBKU is the first one that is not tailored to a certain application scenario. While existing solutions such as the Austrian Mobile Phone Signature have been developed for a specific deployment scenario, the ServerBKU has been designed such that it can be easily integrated into arbitrary application and deployment scenarios. This way, the ServerBKU leverages the use of eID and e-signature functionality in arbitrary applications and helps to improve their provided level of security. In particular, the ServerBKU offers application an attractive alternative to insecure password-based authentication schemes, which will hopefully be history in the future.

Acknowledgements: The authors have been supported by the European Commission Seventh Framework Programme through project *FutureID*, grant agreement number 318424.

REFERENCES

[1] C. Rath, S. Roth, M. Schallar, and T. Zefferer, "A secure and flexible server-based mobile eID and e-signature solution," in Proceedings of the 8th International Conference on Digital Society, ICDS 2014, Barcelona, Spain. IARIA, 2014, pp. 7 – 12.

[2] B. Ives, K. R. Walsh, and H. Schneider, "The domino effect of password reuse," Commun. ACM, vol. 47, no. 4, Apr. 2004, pp. 75–78, [accessed November, 2014]. [Online]. Available: http://doi.acm.org/10.1145/975817.975820

[3] D. Florencio and C. Herley, "A large-scale study of web password habits," in Proceedings of the 16th international conference on world wide web, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 657–666, [accessed November, 2014]. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242661

[4] S. Arora, "National e-id card schemes: a european overview," Inf. Secur. Tech. Rep., vol. 13, no. 2, May 2008, pp. 46–53, [accessed November, 2014]. [Online]. Available: http://dx.doi.org/10.1016/j.istr.2008.08.002

[5] T. Zefferer and V. Krnjic, "Usability evaluation of electronic signature based e-government solutions," in Proceedings of the IADIS International Conference WWW/INTERNET 2012, pp. 227 – 234.

[6] A. Ruiz-Martinez, D. Sanchez-Martinez, M. Martinez-Montesinos, and A. F. Gomez-Skarmeta, "A survey of electronic signature solutions in mobile devices," JTAER, vol. 2, no. 3, 2007, pp. 94–109. [Online]. Available: http://dblp.uni-trier.de/db/journals/jtaer/jtaer2.html#Ruiz-MartinezSMG07

[7] "Handy signatur und buergerkarte," 2014, [retrieved: November, 2014]. [Online]. Available: http://www.buergerkarte.at/

[8] "Estonia eID," 2014, [accessed November, 2014]. [Online]. Available: http://www.id.ee/?lang=en

[9] "Belgium eID," 2014, [accessed November, 2014]. [Online]. Available: http://eid.belgium.be/en/

[10] "Spanish eID," 2014, [accessed November, 2014]. [Online]. Available: http://www.dnielectronico.es/

[11] European Parliament and Council, "Directive 1999/93/ec on a community framework for electronic signatures," December 1999.

[12] European Commission, "Proposal for a regulation of the european parliament and of the council on electronic identification and trust services for electronic transactions in the internal market," 2012. [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2012:0238:FIN:EN:PDF

[13] E. Pisko, "Mobile electronic signatures: progression from mobile service to mobile application unit," in ICMB. IEEE Computer Society, 2007, p. 6. [Online]. Available: http://dblp.uni-trier.de/db/conf/icmb/icmb2007.html#Pisko07

[14] P. Teufl, T. Zefferer, C. Wörgötter, A. Oprisnik, and D. Hein, "Android - on-device detection of sms catchers and sniffers," in International Conference on Privacy & Security in Mobile Systems, 2014, in press.

[15] C. Orthacker, M. Centner, and C. Kittl, "Qualified mobile server signature," in Security and Privacy – Silver Linings in the Cloud, ser. IFIP Advances in Information and Communication Technology, K. Rannenberg, V. Varadharajan, and C. Weber, Eds., vol. 330. Springer Berlin Heidelberg, 2010, p. 103–111. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15257-3_10

[16] "Estonia mobile eID," 2014, [accessed November, 2014]. [Online]. Available: http://mobiil.id.ee/

[17] "Norway eID," 2014, [accessed November, 2014]. [Online]. Available: https://www.bankid.no/

[18] "Austrian Handy Signatur," 2014, [accessed November, 2014]. [Online]. Available: https://www.handy-signatur.at/

[19] K. Stranacher, A. Tauber, T. Zefferer, and B. Zwattendorfer, The austrian identity ecosystem - an e-government experience book title: architectures and protocols for secure information technology, ser. Advances in Information Security, Privacy, and Ethics (AISPE). Antonio Ruiz Martinez, Rafael Marin-Lopez, Fernando Pereniguez-Garcia, 2013, pp. 288 – 309.

[20] H. Leitold, A. Hollosi, and R. Posch, "Security architecture of the austrian citizen card concept," in Proceedings of 18th Annual Computer Security Applications Conference (ACSAC'2002), Las Vegas, 9-13 December 2002. pp. 391-400, IEEE Computer Society, ISBN 0-7695-1828-1, ISSN 1063-9527., 2002.

[21] "Spring Framework," 2014, [accessed November, 2014]. [Online]. Available: http://projects.spring.io/spring-framework/

[22] "Hibernate," 2014, [accessed November, 2014]. [Online]. Available: http://hibernate.org/

[23] "Apache Active MQ," 2014, [accessed November, 2014]. [Online]. Available: http://activemq.apache.org/

[24] "IAIK JCE," 2014, [accessed November, 2014]. [Online]. Available: http://jce.iaik.tugraz.at/

[25] The Open Web Application Security Project, "Owasp top 10 - 2013 the ten most critical web application security risks," 2013, [accessed November, 2014]. [Online]. Available: https://www.owasp.org/index.php/Top_10_2013

[26] PortSwigger Ltd, "Burp suite," [accessed November, 2014]. [Online]. Available: http://portswigger.net/burp/

[27] "Java Server Faces," 2014, [accessed November, 2014]. [Online]. Available: https://javaserverfaces.java.net/

[28] "Primefaces," 2014, [accessed November, 2014]. [Online]. Available: http://www.primefaces.org/

[29] "Spring Roo," 2014, [accessed November, 2014]. [Online]. Available: http://projects.spring.io/spring-roo/

[30] "Spring Web MVC," 2014, [accessed November, 2014]. [Online]. Available: http://docs.spring.io/spring/docs/3.2.x/spring-framework-reference/html/mvc.html

[31] "Prime UI," 2014, [accessed November, 2014]. [Online]. Available: http://www.primefaces.org/primeui/

[32] "handsontable," 2014, [accessed November, 2014]. [Online]. Available: http://handsontable.com/

[33] "vis.js," 2014, [accessed November, 2014]. [Online]. Available: http://visjs.org/

[34] "ServerBKU prototype deployment," 2014, [retrieved: November, 2014]. [Online]. Available: https://pheasant.iaik.tugraz.at:8443/Registration/

No Place to Hide: A Study of Privacy Concerns due to Location Sharing on Geo-Social Networks

Fatma S. Alrayes and Alia I. Abdelmoty

School of Computer Science & Informatics
Cardiff University
Wales, UK
Email: {F.S.Alrayes, A.I.Abdelmoty}@cs.cardiff.ac.uk

Abstract—**User location data collected on Geo-Social Networking applications (GeoSNs) can be used to enhance the services provided by such applications. However, personal location information can potentially be utilised for undesirable purposes that can compromise users' privacy. This paper presents a study of privacy implications of location-based information provision and collection on user awareness and behaviour when using GeoSNs. The dimensions of the problem are analysed and used to guide an analytical study of some representative data sets from such applications. The results of the data analysis demonstrate the extent of potential personal information that may be derived from the location information. In addition, a survey is undertaken to examine user awareness, concerns and subsequent attitude and behaviour given knowledge of the possible derived information. The results clearly demonstrate users' needs for improving their knowledge, access and visibility of their data sets as well as for means to control and manage their location data. Future work needs to investigate the current state of personal data management on GeoSNs and how their interfaces may be improved to satisfy the highlighted users' needs and to protect their privacy.**

***Keywords*–location privacy; Geo-social networks; mobility patterns; privacy concerns.**

I. Introduction

The proliferation and affordablity of GPS-enabled devices are enabling individuals to accumulate an increasing amount of personal information, such as their mobility tracks, geographically tagged photos and events. Embracing these new location-aware capabilities by social networks has led to the emergence of Geo-Social Networks (GeoSNs) that offer their users the ability to geo-reference their submissions and to share their location with other users. Subsequently, users can use location identifiers to browse and search for resources. GeoSNs include Location-Enabled Social Networks (LESNs), for example, Facebook, Twitter, Instagram and Flickr, where users' locations are supplementary identification of other primary data sets, and Location-Based Social Networks (LBSNs), for example, Foursquare and Yelp, where location is an essential key for providing the service.

In addition to location data that describe the place the places visited by users, GeoSNs also records other personal information, such as user's friends, reviews and tips, possibly over long periods of time. User's historical location information can be related to contextual and semantic information publicly available online and can be used to infer personal information and to construct a comprehensive user profile [1], [2]. Derived information in such profiles can include user activities, interests and mobility patterns [3], [4]. Such enriched location-based profiles can be considered to be useful if used to personalise and enhance the quality of the services provided by the social networking applications. For example, by recommending a place to visit on Foursquare and showing local trends on Twitter. However, they can potentially be used for undesirable purposes and can pose privacy threats ranging from location-based spams to possible threats by an adversary [5]. Users may not be fully aware of what location information are being collected, how the information are used and by whom, and hence can fail to appreciate the possible potential risks of disclosing their location information.

In this paper, a study of location privacy of users when using GeoSNs is presented. The aims are to investigate potential privacy implications of GeoSNs, as well as examine users' privacy concerns and attitude when using these networks. We demonstrate the privacy implications by identifying possible derived information from typical data sets collected by LBSNs for different types of users, as was shown in an earlier work [1]. In addition, a survey was undertaken to gauge users' understanding and reaction to possible types of privacy threats resulting from the knowledge of their location information.

Firstly, the dimensions of the problem are examined and the factors that can impact users' privacy are identified. These factors include, the type of data collected, its visibility and accessibility by users, as well as the possible exploitation of these data by the application. Secondly, an analytical study is conducted using a representative data set to explore the location data content and the range of possible inferences that can be made from them. The frequency of usage of the networking application is used to classify users and in the analysis of their behavioural patterns. Finally, a survey was undertaken to examine users' awareness and concerns with respect to privacy implications of their location data and their needs to control access to their data on GeoSNs. Previous studies explored users' privacy concerns and attitude when sharing their location for social purposes, but presented limited evaluations using restricted application scenarios [6], [7]. Questionnaire analysis demonstrate a strong feasibility of inference of users' personal information that may pose a threat to their privacy on these networks. The survey also reveals users' concerns about their location privacy and their motivation to control their location information. The outcomes highlight the need for further work on improving the visibility of the information collected, to allow users to better understand the implications of their location sharing activities and assess their need to control access to their location data sets.

The rest of this work is organized as follows. Section II gives an overview of related work. In Section III, the dimensions of the location privacy problem in GeoSNs are discussed. Section IV describes the experiment conducted with a realistic data set to explore the spatiotemporal information content explicitly described and that may be inferred from the data. Section V builds on the results of Section IV by designing and deploying a questionnaire that explores users' awareness and attitude towards potential privacy threats. Discussion of the results and conclusions are presented in Section VI.

II. Related Work

Security and privacy of online social networks is a general research area that includes evaluating potential privacy risks, as well as developing privacy-protection methods [8], [9], [10]. This paper focuses on the privacy implications of location-related information in GeoSNs. Two relevant questions to the problem studied are: to what extent is location privacy a potential concern for users in GeoSNs, and what sort of location-based inference is possible from the data collected in GeoSNs. In this section, related works on both issues are reviewed.

A. Users' Attitude and Privacy Concerns in Geo-Social Networks

Much interest has been witnessed over the past few years for studying users' attitude and concerns to location privacy and investigating how user-empowered location privacy protection mechanisms can influence their behaviour. Tsai et al. [6] developed a social location sharing application, where participants were capable of specifying time-based rules to share their location and were then notified of who viewed their locations. Their findings suggested that the control given to users for setting their sharing preferences contribute to the reduction of the level of their privacy concern.

Sadeh et al. [7] enabled users of their *People Finder* application to set rule-based location privacy controls by determining the where, when and with whom to share their location and were notified when their location information was requested. Participants were initially reluctant to share their location information and then tended to be more comfortable over time. Patil et al. [11] developed a system to represent actual users' workplace, offering live feeds about users and their location and asked users to define different levels of permissions for their personal information sharing. They found that participants were concerned most about their location information and that they utilised the permission feature to control this information. Another study by Kelley et al. [12] showed that users were highly concerned about their privacy especially when sharing location information with corporate-oriented parties.

Other works were carried out to examine how the employment of visualization methods may impact users' attitude to location privacy and behaviour. Brush et al. [13] studied users' attitude towards their location privacy when using GPS tracking over long periods of time and questioned whether using some obfuscation techniques can address their concerns. Participants were concerned about revealing their home, identity and exact locations. They visually recognised and chose the best obfuscation techniques they felt can protect their location privacy. In addition, Tang et al. [14] investigated the extent of presenting various visualizations of users' location history on influencing their privacy concerns when using location-sharing applications. They developed text-, map-, and time-based visualization methods and considered spatiotemporal properties of sharing historical location. They noted that the majority of participants found visualization of location history to be more revealing and tended to prefer text-based presentation methods to limit the amount of data exposed.

With regards to public GeoSNs, there are relatively few research works that examine privacy concerns of users. Lindqvist et al. [15] considered users' motivations in using Foursquare and questioned their privacy concerns. Their analysis showed that most of the participants had few concerns about their privacy and users who were more concerned about their privacy chose not to check into their private residence or to delay checking into places till after they leave, as a way of controlling their safety and privacy. A similar observation was noted by Jin et al. [16], where it was found that users were generally aware of the privacy of their place of residence and tended not to provide full home addresses and blocked access to their residential check-ins to other users.

In summary, it is evident that location privacy presents a real concern to users in location-sharing applications, and particularly as they become aware of the data they are providing. Previous studies may have been limited by several factors, including the size and representativeness of the sample user base used in the experiments conducted and the limited features of the proprietary applications used in testing [6], [7], [11], [12]. Moreover, as far as we are aware, no studies so far have considered the problem of location privacy on public LBSNs.

B. Location-Based Inference from GeoSNs

There are some studies that utilised publicly available information from GeoSNs in order to derive or predict users' location. In [17], Twitter users' city-level locations were estimated by only exploiting their tweet contents with which it was possible to predict more than half of the sample within 100 miles of their actual place. Similarly, Pontes et al. [18] examined how much personal information can be inferred from the publicly available information of Foursquare users and found the home cities of more than two-thirds of the sample within 50 kilometres. Sadilek et al. [19] investigated novel approaches for inferring users' location at any given time by taking advantage of knowing the GPS positions of their friends on Twitter. Up to 84% of users' exact dynamic locations were derived. Interestingly, Gao et al. [20] formulated predictive probability of the next check-in location by exploiting social-historical ties of some Foursquare users. They were able to predict with high accuracy possible new check-ins for places that users have not visited before by exploiting the correlation between their social network information and geographical distance in LBSNs [21].

Other works focussed on investigating the potential inference of social relationships between users of GeoSNs. Crandall et al. [22] investigated how social ties between people can be derived from spatial and temporal co-occurrence by using publicly available data of geo-tagged pictures from Flickr. They found that relatively limited co-occurrence between users is sufficient for inferring high probability of social ties. Sadilek et al. [19] also formulated friendship predictions that derive social relationships by considering friendship formation patterns, content of messages of users and their location. They predicted 90% of friendships with accuracy beyond 80%. Additionally, Scellato et al. [23] investigated the spatial properties of social networks existing among users of three popular LBSNs and found that the likelihood of having social connection decrease with distance. In [24], they developed a link prediction system for LBSNs by utilising users' check-ins information and properties of places. 43% of all new links appeared between users with at least one check-in place in common and especially for those who have a friend in common.

Studying and extracting spatiotemporal movement and activity patterns of users on GeoSNs attracted much research in recent years. Dearman et al. [25] exploited location reviews on Yelp in order to identify a collection of potential activities promoted by the reviewed location. They derived the activities supported by each location by processing the review text and validated their findings through a questionnaire. Noulas et al. [26] studied user mobility patterns in Foursquare by considering popular places and transitions between place categories. Cheng et al. [27] examined a large scale data set of users and their check-ins to analyse human movement patterns in terms of spatiotemporal, social and textual information associated with this data. They were able to measure user displacement between consecutive check-ins, distance between users' check-ins and their centre of mass, as well as the returning probability to venues. They also studied factors affecting users' movement and found considerable relationship between users' mobility and geographic and economic conditions. More recently, Preotiuc-Pietro et al. [28] investigated the behaviour of thousands of frequent Foursquare users. They analysed users' movements including returning probability, check-in frequency, inter-event time, and place transition among each venue category. They were also able to group users based on their check-in behaviour such as generic, businessmen or workaholics as well as predict users' future movement. The above studies show a significant potential for deriving personal information form GeoSNs and hence also imply the possible privacy threats to user of these applications. Whereas previous studies considered mobility and behaviour of large user groups and determined general patterns and collective behaviour, in this work we consider the privacy implications for individual users, with the aim of understanding possible implied user profiles from location data stored in GeoSNs.

III. Dimensions of the Location Privacy Problem on GEOSNS

Four aspects of the data collected can be identified that can affect location privacy. These are: 1) the amount of data collected and its quality, 2) its visibility and accessibility, 3) its possible utilisation by potential users, and 4) the level of security offered to users by the application. This discussion focuses on the type of privacy-related questions that can be asked and the confidence level in the information that can be derived. Both factors can affect the degree of privacy concern to users. The study considers both LBSNs (Foursquare) and LESNs (Twitter), the difference in the way location data are acquired in both and the issues implied.

A. Location Data Collection

Here the types of data, its density and quality, as well as the methods of collection and storage are considered.

1) Method of Collection: Both LBSNs and LESNs depend on the user device to acquire the user's current location using GPS, wireless access points (WAP) or cellular networks. When using LBSNs, location data are collected automatically since location is mandatory to providing the service. In Foursquare specifically, user's location is implicitly acquired on a continuous basis, even without using the service. User's check-ins into specific places are verified against their estimated current location and recorded explicitly. In LESNs, user's location data are collected only when location-based features are enabled and used. Some features require continuous collection of location data, for example, when tailoring trends to the user's location in Twitter. The mode of data collection, whether continuous or periodic; automatic or manual, will impact the volume of data collected and its accuracy, and hence also the degree of confidence in inferences made from the data.

2) Types of Data: The completeness and accuracy of location information are primary factors that determine the possible inferences made based on this information and the possible privacy threats to users. Three types of data can be associated with location data collected in GeoSNs: spatial, non-spatial and temporal.

- Spatial semantics: These refer to any type of information that can be used to identify the places visited. In both LBSNs and LESNs, user's location is identified as a point in space with a latitude and longitude. In LBSNs, users identify their locations explicitly, allowing for a rich definition of place identity, including place name, type classification and street address. On the other hand, location in LESNs is determined automatically by reverse geocoding the registered latitude and longitude coordinates, and thus carry a degree of inaccuracy and ambiguity. Increasingly, some LESNs are able to use resources from LBSNs for defining locations. For instance, Instagram allows users to geotag their pictures using the Foursquare API [29]. Twitter also uses Google API for linking users' selected place names with a location on a map. Hence, in both cases it can be assumed that detailed and precise place identities visited by users may be stored by the applications.
- Non-spatial semantics: Non-spatial semantics are other types of data about both users and places that may be associated with location information. These include explicit user data, as for example defined in their personal profiles on the application or place-related data, such as reviews, tags and pictures. With the user permission, applications will identify users and share their personal information. Rich place-related semantics may also be mined from resources on the web [30].
- Temporal semantics: These represent the time of user's visit to a place and the duration of their visit. In LBSN, the time of visit is registered by the user as they

check-in to a place. The user's physical presence in the place may be validated by comparing their actual GPS coordinates with those of the place they check into. In LESN, a time stamp is encoded with the resource used, for example, a tweet location. However, in this case it is difficult to ascertain whether the user is intentionally visiting the place or happened to be passing by it. In both cases, further processing of the user tracks is needed to estimate the duration of the user's visit.

3) Data Volume: The amount of location data collected is another important factor to be considered and is dependent of the user attitude and behaviour when using the application. The pattern of data logging and the frequency of usage will determine the density of the data collected over time and will thus influence the type of information that may be inferred from the data. For example, regular visits to specific places can determine routine mobility patterns, while incidental visits to other places can signify special events or activities.

B. Location Information Accessibility

Location information accessibility represents how much of the user's data are available and visible to others including the user, other users and third parties of the service. In terms of users' accessibility to their collected location and location-related data, GeoSNs provides only limited means for accessing these kinds of information. In Foursquare, users' previous check-in information are available in the form of check-in history, where users can view their visited venues, dates of visits and tips they made. These raw data provide only a limited view of the information content in the data, as discussed in the previous section. In Twitter, users can request to download their tweet history, but location information are not included in this data. As for information visibility, most of the users' information published on GeoSNs are available to their friends and can be visible to other users.

Generally, users of GeoSNs have limited control over the visibility and accessibility of their information by others, since the privacy settings provided to them is not adequate enough to manage all aspects of their information accessibility. In Foursquare, almost all of the user's information is publicly available by default and can be viewed by other users. This include profile information, tips, likes, friends list, photos, badges, mayorships, and check-ins. Users are only able to block access to their check-ins and photos by setting their view to private. Similarly in Twitter, users' profiles and their tweets are public by default, and can be accessed by others. This means that location information attached with tweets is publicly available as well unless users mark their profile as private, where only followers can view their data. All of the publicly available users' information is accessible by third parties including the geo-social application APIs users. Third parties can also have privileges to access the user's personal information. In the case of Foursquare, third parties can get check-in data in anonymous form, but they also indicate that they will share user's personal information with their business partners and whenever is necessary in some situations, such as enforcement of law. Twitter, on the other hand, states that any content the user submits or displays through the service is available to their third parties without anonymity.

C. Location Data Exploitation

Location information exploitation refers to how the application or third parties can utilise the data and for which purposes. This dimension involves the actual exploitation of user's location and location-related data that lead to posing various levels of privacy threats. It seems that GeoSNs have unlimited rights to utilise their users data in any way, for any purpose as stated in their terms of use. For example, Foursquare gives itself absolute privileges over using and manipulating user information as stated in their terms of use [31].

"By submitting User Submissions on the Site or otherwise through the Service, you hereby do and shall grant Foursquare a worldwide, non-exclusive, royalty-free, fully paid, sublicensable and transferable license to use, copy, edit, modify, reproduce, distribute, prepare derivative works of, display, perform, and otherwise fully exploit the User Submissions in connection with the Site, the Service and Foursquare's (and its successors and assigns') business, including without limitation for promoting and redistributing part or all of the Site (and derivative works thereof) or the Service in any media formats and through any media channels (including, without limitation, third party websites and feeds)."

Similarly, Twitter has the right to utilise users data, including location information, in various ways, as stated in their terms of use [32].

"By submitting, posting or displaying Content on or through the Services, you grant us a worldwide, non-exclusive, royalty-free license (with the right to sublicense) to use, copy, reproduce, process, adapt, modify, publish, transmit, display and distribute such Content in any and all media or distribution methods."

It is clear therefore that there are no commitments on GeoSNs as to how the data may be used or shared by the application or by other parties. In addition, the reasons for the potential exploitation of users data are vague (e.g., to improve the services) or even not stated. Hence, by agreeing to the terms and conditions, users effectively are giving away their data and unconditional rights to the use of their data to the application.

D. Location Data Security

Location data security refers to the level of data protection provided by the application for securing the user's data against the risk of loss or unauthorized access. In general, the fact that data are stored somewhere on servers opens the doors for potential undeclared access and use, and hence it is almost impossible to guarantee the security of the user data. Foursquare declares that the security of users' information is not guaranteed and any "unauthorized entry or use, hardware or software failure, and other factors, may compromise the security of user information at any time". Without any commitment to responsibility for data security, the application provider is declaring the possible high risk of data abuse by any adversary or even by the application provider themselves. Twitter states that "Twitter complies with the U.S.-E.U. and U.S.-Swiss Safe Harbor Privacy Principles of notice, choice, onward transfer, security, data integrity, access, and enforcement", but give no additional explanation or examples on situations or access methods that these laws apply to.

In the following section, a sample data set from a LBSN is used to explore and analyse the potential information content that can be derived from the location data.

IV. Empirical Investigation

This analysis is carried out using a real-world data set from Foursquare, as a typical example of a LBSN. The purpose is to demonstrate possible privacy implications in terms of personal information inferences and exploitation from user activity on GeoSNs. The effect of location data density and diversity on the possible inferences that can be made is analysed.

A. Dataset

The Foursquare dataset used in this analysis is provided by Jin et al. [16]. The dataset contains venue information and public check-ins for anonymised users around the wide area of Pittsburgh, USA from 24 February, 2012 to 22 July, 2012. Places on Foursquare are associated with pre-defined and structured place categories, e.g., Home, Office, Restaurant, etc. The data set contains 60,853 local venues, 45,289 users and 1,276,988 public check-ins of these users.

B. Approach and Tools Used

To study the possible impact of location data density on users' privacy, users of the dataset were first classified into groups based on their check-in frequency. A filter was initially imposed to disregard sparse user activity. Hence, users with less than five check-ins per month were removed from the dataset. The rest of the users were categorised into three groups based on their check-in frequency per day, to moderate, frequent and hyper-active user groups, as shown in Table I. One representative user is selected from each group who has the nearest average check-ins per day to the average check-ins for the whole group. Table II shows some statistics for the selected users. The R statistical package was used for analyses and presentation of results. Mainly, the SQLDF package was used for querying, linking and manipulating the data and the ggplot2 package was used for the presentation of the results of the analysis [33].

C. Results

Analysis of the data set questioned the sort of implicit user-related information that can be considered to be private that may be extracted using the location data collected. User's spatial location history can be extracted in the form of visits to venues and the exact times of such visits. The places visited are identified and described in detail. For example, *user7105* visited 'Kohl's'; a department store, located at latitude 40.5111 and longitude -79.9934 at 9 a.m. on Monday 27/2/2012. The basic information on venue check-ins can be analysed further and combined with other semantic information from the user profile to extract further information that can compromise user's privacy. Analysis will investigate the relationship between users and places visited, their mobility patterns and the relationships between users and other users as follows.

TABLE I: Statistics of user groups in the Foursquare dataset.

Group Name	Check-ins Range in Total	Users Count	Check-ins Range per Day	Average Check-ins per Day
Moderate	Between 50 and 300	4902	0.3 to 2	1.15
Frequent	Between 301 and 750	880	2 to 5	3.5
Hyper-active	Between 751 and 1303	24	5 to 8.6	6.8

TABLE II: Profiles of selected users.

Factor	Selected Users		
	User9119	*User7105*	*User2651*
Number of total check-ins	144	511	1019
Average check-ins per day	0.96	3.4	6.8
Number of visited venues	21	99	101
Number of visited venues categories	17	47	57
Number of visited venues main categories	10	11	17
Number of friends	20	10	19

- Degree of association between user and place. Relationship with individual place instances as well as with general place types or categories will be studied. Elements of interest will include visit frequency, and possible commuting habits in terms of the association between the visit frequency of places and their location.
- Spatiotemporal movement patterns. Visiting patterns to individual places or to groups of places can identify regular movement patterns. In addition, a change of visit patterns can also be a significant pointer to user activity.
- Degree of association with other users. Relationship between users can be derived by studying their movement patterns and analysing their co-occurrence in place and time.

1) The Moderate User: The analysis results of *user9119* selected from the moderate group are as follows.

a) Degree of Association Between User and Place: Two frequently visited venues by *user9119* are 'Penn Garrison' whose category is 'Home' and 'USX Tower' whose category is 'Office' representing 44% and 36%, respectively of the total check-ins. Home and Office are highly sensitive places, yet they represent 80% of this user's check-ins. Other visited place types with significantly less frequency include, 'Nightlife Spot': 0.5%, 'Travel & Transport': 0.27%, and 'Shop & Service': 0.27%. *User9119* is also interested in 'Hockey', 'Garden Center' and 'Museum' place types. As could be predicted, the location of venues visited indicates that most of them are close to 'Home' and 'Office', whereas this user commutes further away to visit some less frequent venues such as 'Hockey Arena'. Figure 1 shows this user's check-in frequency for different categories of venues classified by the time of day. As can be seen from the figure, this user's association with sensitive places like home and place of work can be identified. In addition, a strong association with other place categories is also evident.

b) Spatiotemporal Movement Patterns: About 40% of this user's total check-ins occurs at 9 am, mostly in the 'Office' and at 7pm, mostly at 'Home'. More than two-thirds of the check-ins are between 10am and 2pm and between 6pm and 11pm, which indicates that this user commutes more frequently during these hours. From the weekly patterns of movement, it can be seen that 71% of the venues were visited after 6pm. Mondays and Thursdays are when this user is most active, representing 41% of the check-ins. *User9119* tends to go to 'Nightlife spots' more frequently during working days, whereas visits to other specific place types occur only at weekends, including, 'Salon or Barbershop', 'Coffee Shop' and 'Garden Centre'. This user

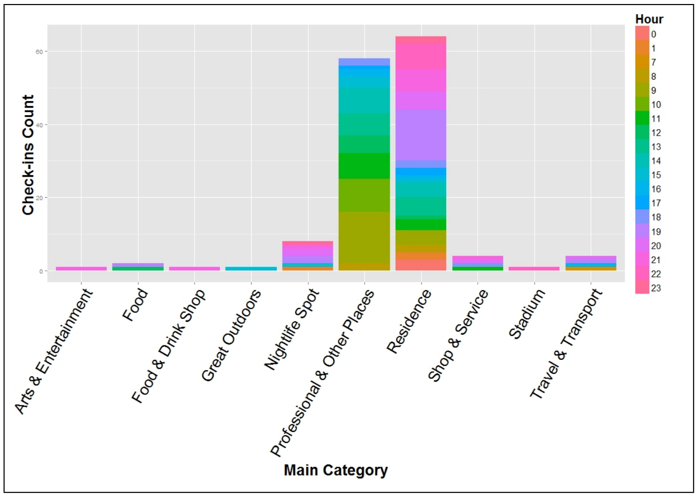


Figure 1: The moderate user's check-ins count, classified by the category of venues for different hours of the day.

typically starts commuting earlier on working days and visits more places than on weekends. Observing the check-ins by month shows that the months of May and June are the most active in terms of check-in frequency, comprising 60% of total check-ins, as well as diversity of category of venues visited (99% of the total visited categories of venues occurred in those months, including the emergence of new categories such as 'Museum', 'Airport' and 'Hotel'). The user was least active in April. Figure 2 demonstrates this user's check-ins count in different categories of venues, classified by day and grouped by month. Some changes of this user's habits can be noticed as well, which can suggest a change of personal circumstances. For example, the user has not visited any Nightlife spots in March and April and has not checked-in in any place on Sundays of June and July including 'Home' and 'Office'. In addition, the user has not checked in any place for a period of a week between the 21st and 28th of April. *User9119* last check-in before this week was on the 20th of April at 'Home'. This may indicate a possible period of time-off work in that week.

c) Degree of Association with Other Users: Co-location is used here to denote that users have visited the same venue at the same time. This can be used as a measure of interest in a place and relationships between users. *User9119* was co-located in 6 unique venue categories with two (out of twenty) friends. He shared three co-occurrences with two friends; once with *friend1236* at 'American Restaurant' and twice with *friend15229* at 'Office', which may indicate that *friend15229* is a colleague at work. In fact, this user shared 95 co-occurrences with 52 other users, 90% of which were in the 'Office' suggesting the probability of those users being work colleagues.

2) The Frequent User: Analysis of results of *user7105* from the frequent user group is as follows.

a) Degree of Association between User and Place: Similar to the moderate user, *user7105* most checked-in venue category is 'Home', whose location is identified in detail. However, the second most visited venue is a specific restaurant, whose category is 'American Restaurant', representing 25% of the total check-ins and 28% of category check-ins. This visit pattern may indicate that this is the user's work place. The third most visited venue category for this user is 'Bar' (4%), that is a subcategory of 'Nightlife Spot', representing about 7% of check-ins. Generally, the third most visited main category is 'Shop & Service' corresponding to 10% of check-ins, where specifically 40% of those are to 'Gas Station or Garage' and 25% are to 'Drugstore or Pharmacy'. *User7105* is occasionally interested in visiting places described as 'Great Outdoors', 'Professional & Other Places' and 'Arts & Entertainment'.
The majority of the most frequently visited venues are within close distance to 'Home' and to the 'American Restaurant', whereas *user7105* commutes further away for other less frequently visited places, such as, 'Medical Center'.

b) Spatiotemporal Movement Patterns: Generally, about 20% of the check-ins occurs from 10am to 12pm, half of which are at 'Home'. In addition, *user7105* tends to move the most between 3pm and 5pm, representing 23% of his total check-ins to 46% of the visited venues' categories. More than half of the check-ins are at 'Atria's', which may indicate that the user starts his work shift in this place at that time. This hypothesis can be ascertained by examining his subsequent check-ins, where 18% of the check-in happens between 12am and 3am at 'Home', possibly when the user comes back from work. There is a high correlation in terms of place transition between 'Home' and the 'American Restaurant'. When examining the weekly mobility, *user7105* is more active on Tuesdays followed by Saturdays corresponding to 19% and 16%, respectively of total check-ins. Noticeably, the majority of Friday and Tuesday check-ins occurs at 12am, whereas Monday and Saturday at 4pm. Furthermore, this user has visited more diverse venues on Tuesdays followed by Thursdays and Wednesdays representing 53%, 43% and 38%, respectively of the total visited categories. During the working week, this user tends to visit a 'Bar' (5%), especially on Tuesdays, and 'Gas Station or Garage' (4%). This is reasonable considering his working shifts. While on weekends, 'Grocery or Supermarket' and 'Drugstore or Pharmacy' venues are among the top four visited categories corresponding to 4% and 5%, respectively of weekends' check-ins. *User7105s* check-in patterns were regular over the whole period. However, visits of this user are more frequent and diversified in the month of March. Noticeably, about 28% of the check-ins between 12am and 3am occurred in March, indicating a possible change of lifestyle. Figure 3 presents this user's check-ins count in different categories of venues, classified by day and grouped by month.

c) Degree of Association with Other Users: *User7105* had co-locations in 36 unique venues from 19 different categories with 7 friends. In particular, 26 co-locations are shared with *freind38466* at 14 venues categories including 'Coffee Shop', 'Bar', 'Fast Food Restaurant' and 'Other Nightlife'. Co-locations shared with the rest of the friends include 'Bar', 'Mexican Restaurant', 'Hospital' and 'Government Building'. Moreover, *user7105* has 16 spatiotemporal co-occurrences at 14 unique venues from 6 different categories with two friends, where 14 co-occurrences with *freind38466* at 6 different categories including mostly 'Bar', 'American Restaurant', and 'Sandwich Place', which

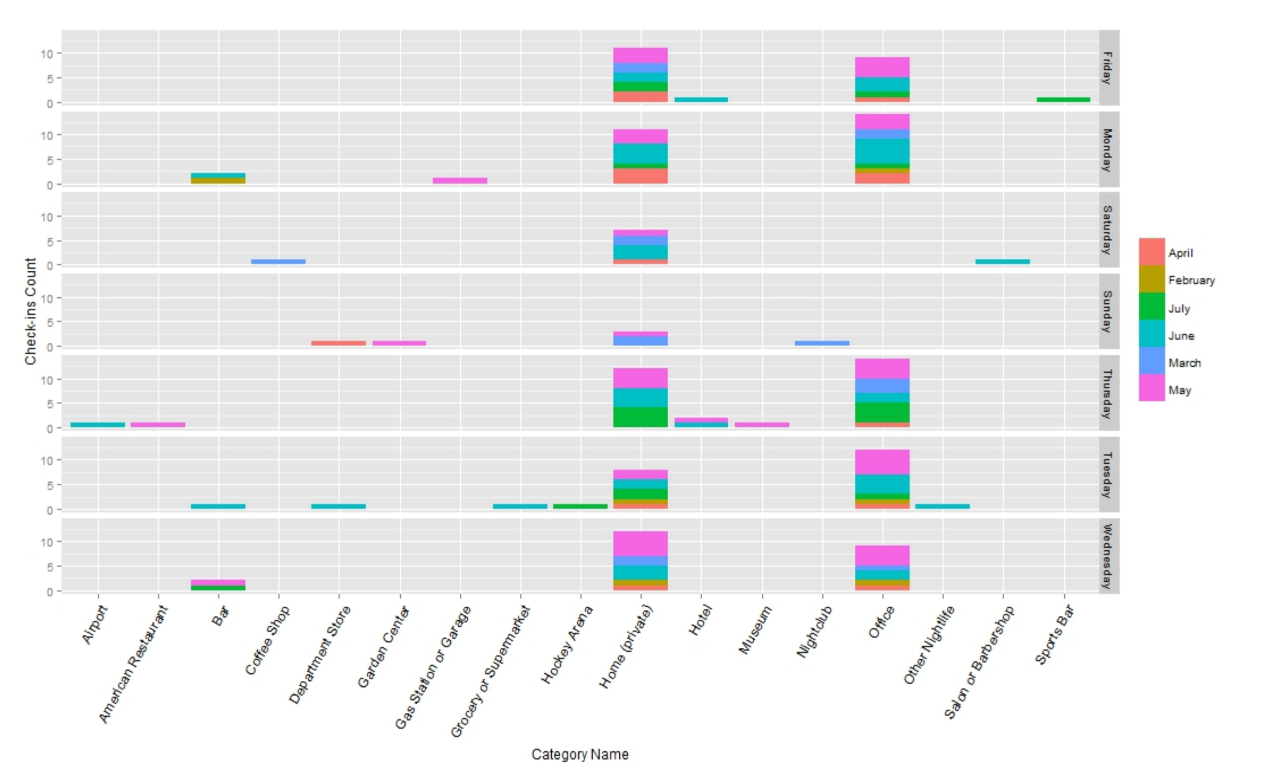


Figure 2: The moderate user's check-ins count in different categories of venues, classified by day and grouped by month.

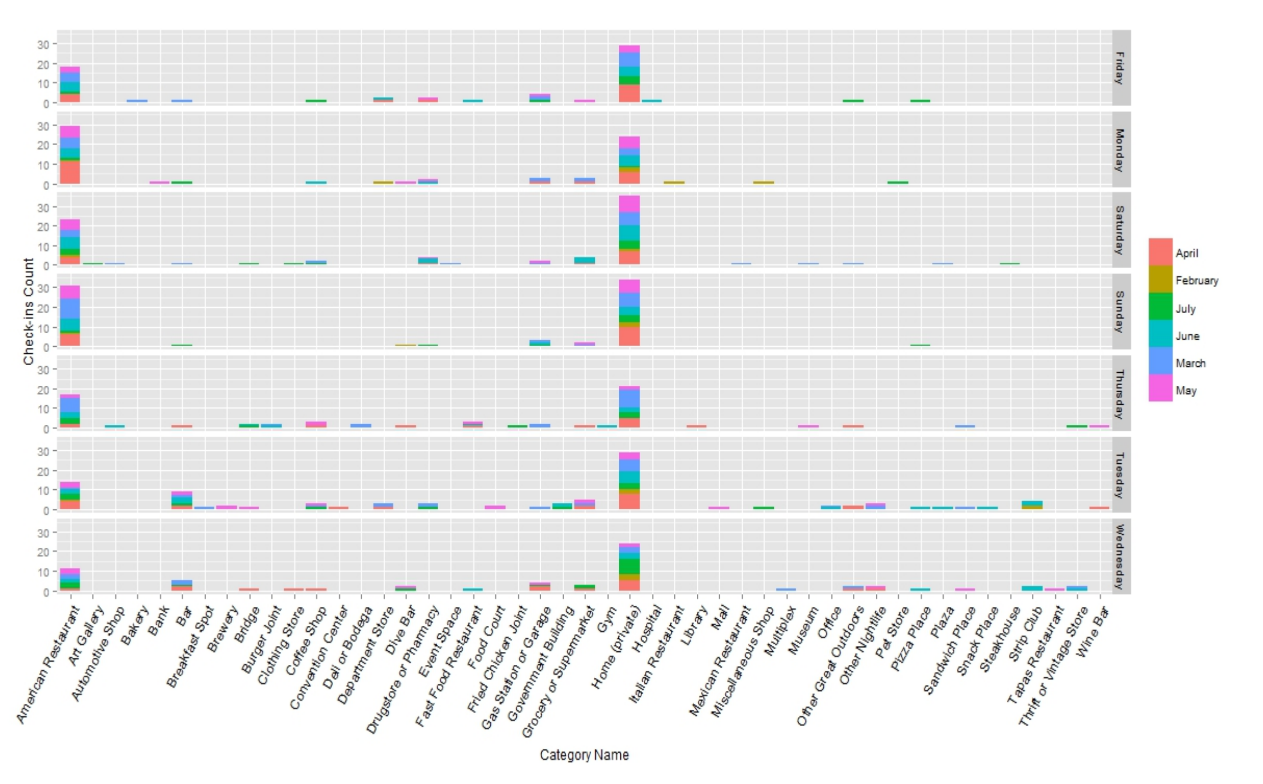


Figure 3: The frequent user's check-ins count in different categories of venues, classified by day and grouped by month.

can denote a close friendship between them. The other two co-occurrences are with *friend15995* at 'American Restaurant' on May 13th and June 17th, 2012. The place and time of this user's co-occurrences with friends are shown in Figure 4. Similarly, this user has 89 co-occurrences with other users, who are not stated as friends, at 29 unique venues, where 38% of these co-occurrences are at 'American Restaurant' and 24% at 'Plaza'.

3) The Hyper-Active User: The results of analysis for *user2651* selected from the hyper-active user group are as follows.

a) Degree of Association Between User and Place: The first most visited venue by this user is a 'Nightlife Spot' corresponding to 15% of total check-ins. Two 'Home' venues were recorded, 'My Back Yard' and 'La Couch', representing 23% of the check-ins. Both home venues have the same location coordinates, implying that they are actually the same place. 'Automotive Shop', 'Pool' and 'Italian Restaurant', representing 9%, 8% and 5%, respectively of this user's total check-ins indicate the user's interests and activities - swimming and Italian food in this case. A particular instance with a vague category of 'Building' was among the top 10 most visited venues. Further investigation of this venue using the given place name revealed that this building is a place where an international summit for creative people is held [34], which may indicate that *user2651* is possibly an active participant of such an event. When considering the main category of the visited venues, this user generally visits 'Shop & Service', 'Nightlife Spot', 'Arts & Entertainment' and 'Food' on a regular basis, representing 17%, 14%, 11% and 10%, respectively of this user's check-ins. *User2651* also usually visits 'Gas Station or Garage': 4%, and 'Church': 3%. The location of the visited venues can be clustered into two main areas on a map as illustrated in Figure 5. One area includes 'Home' as well as other frequently visited venues such as 'Nightlife Spots' and 'Gym or Fitness Center'. The other area includes mostly less frequently visited venues such as 'Hospital'.

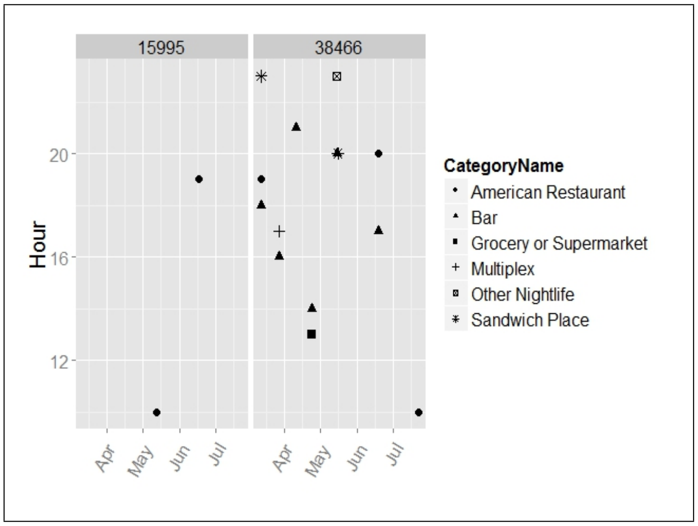


Figure 4: Spatiotemporal tracks of the frequent user co-occurrences with friends.

b) Spatiotemporal Movement Patterns: Overall, 53% of residential check-ins occurs between 9am and 12pm. A significant number of check-ins (10%) occur at 2pm, of which almost two-thirds occur in an 'Automotive Shop'. Check-in frequency reaches another peak between 11pm and 12am (18%), of which more than half are in 'Nightlife Spot'. Noticeably, this user tends to be more active at night, where about 70% of the check-ins are registered after 6pm. In his case, weekends have similar check-in frequency as the working week, but Sundays register as the most active day in terms of check-in frequency. Moreover, *user2651* checks in considerably less frequently at the 'Automotive Shop' and the 'Pool' on Wednesdays and Fridays, but checks in the 'Automotive Shop' and 'Nightlife Spot' in weekends. This may indicate that he works shifts on weekends. *User2651* has regular check-in patterns over the whole period. However, in the months of June and July, check-ins into 'Hotel' and 'Pool' significantly increased representing 75% and 60%, respectively of these venues total check-ins. Figure 6 demonstrates this user's check-ins count in different categories of venues, classified by day and grouped by month.

c) Degree of Association with Other Users: As with other users, *user2651* was co-located with 23 users at 12 distinct venues, half of these co-occurrences happened in 'Bar', 'Automotive Shop' and 'Grocery or Supermarket'. *User2651* is co-located in 27 unique venues from 19 categories with 9 friends, 13 of which are with friend12432 and 9 with friend12046. Most of the co-locations are in 'Nightlife Spots', 'Gas Station or Garage', 'Pool', 'Flower Shop' and 'Bar'.

The three dimensions analysed above will form the basis of the questionnaire design described in the next section.

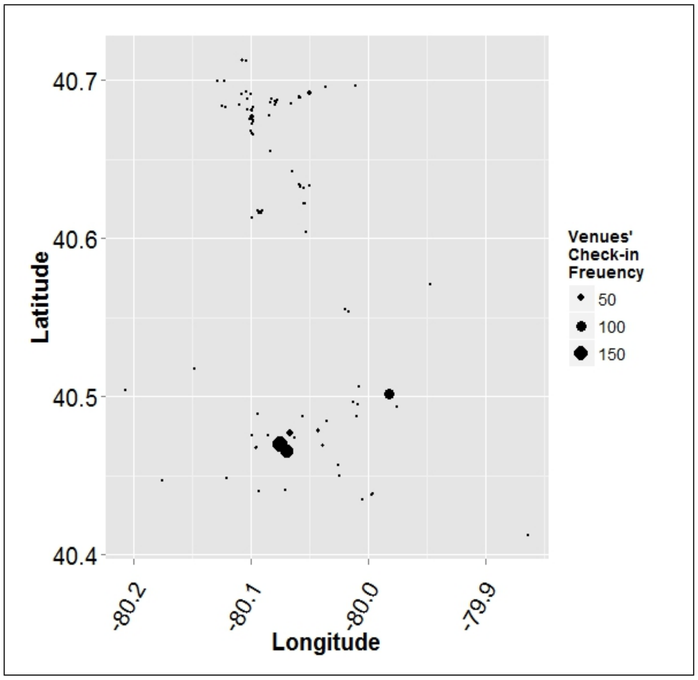


Figure 5: Coordinates of venues visited by the hyper-active user, considering the frequency of visit.

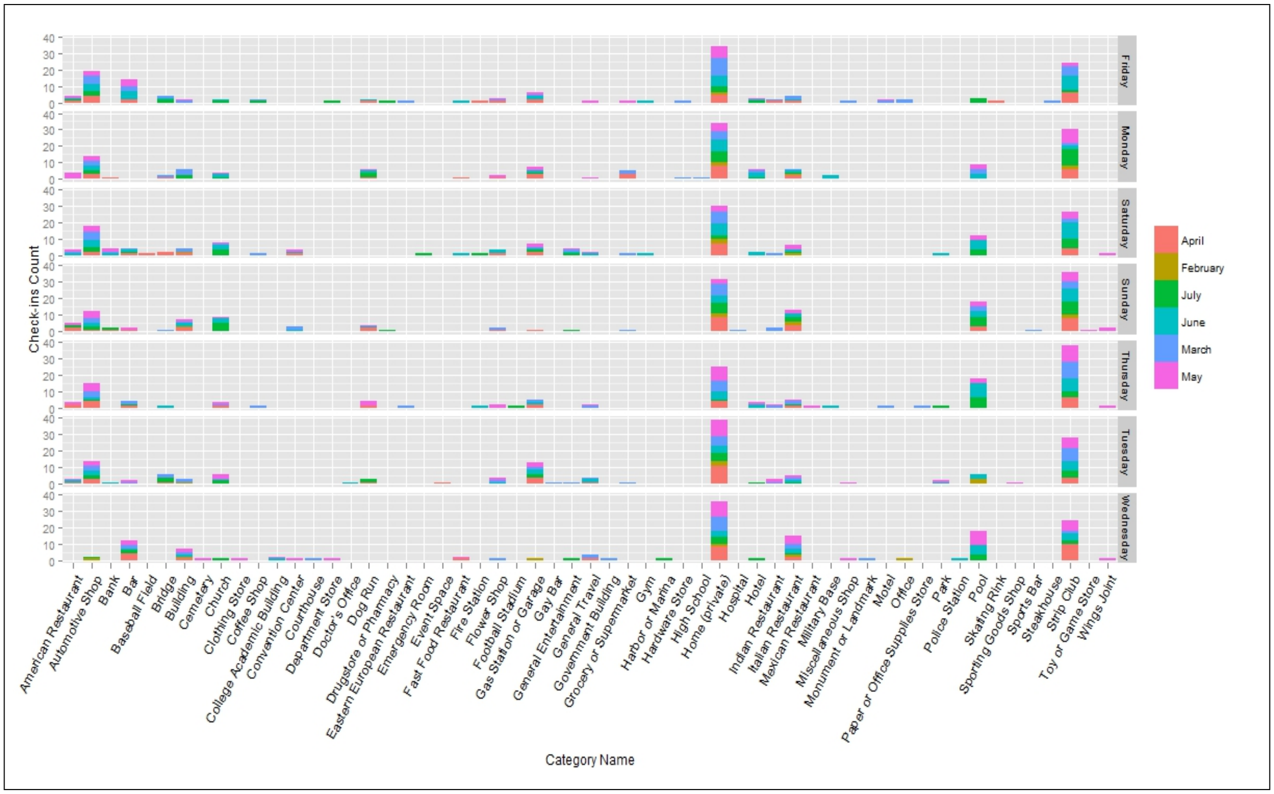


Figure 6: Count of check-ins for the hyper-active user in different categories of venues, classified by day and grouped by month.

V. User Study

None of the related studies reviewed in Section II above has fully explored or focused on improving users' full awareness and understandability of the potential privacy implications when sharing their location information on GeoSNs. Here, a survey is undertaken to examine the privacy concerns and behaviour of users of online social networks, in particular users' concerns towards their location information. Three main aspects are addressed in this study: the extent of users' awareness of the terms of use they sign up to when using these applications, their understanding and attitude to potential privacy implications, and how they may wish to control access to their personal information on these applications.

A. Study Design

The questionnaire was developed using Google Forms. Targeted participants were users of online social networking applications who use location features, e.g., adding location to their posts and photos and checking-in when visiting places. A pilot study was first carried out to ensure the clarity and coherence of the survey. Four volunteers with no specific background completed the survey and provided valuable feedback into the wordings and layout of the questions used. The survey was then disseminated widely within the university to staff and students and was also advertised on social networks through the author's account. A token incentive of £10 Amazon vouchers was offered to ten randomly chosen participants who completed the survey.

The questionnaire consists of four main sections. The first section collects background information on the participants and their use of GeoSNs. The next section examines users' knowledge of terms of use and privacy policies of the applications, followed by a section on studying perception of possible inferences of personal information. The last section is intended to capture users' attitude to privacy on social networks as well as their attitude to controlling their personal information.

B. Results

The questionnaire data were analysed using the R statistical package and the results are presented below. 186 participants completed the survey of which 60% are young adults in the age group 15-24, divided almost equally between males and females. The vast majority of participants (77%) use the services frequently (several times a day) and 72% of participants use the location services in GeoSNs. About 60% use location features on only one application. Adding locations to posts and pictures on Facebook was the most used application, corresponding to 47% of the total number of location services used. This is followed by adding location to tweets on Twitter, photo mapping pictures on Instagram, and checking-in on Foursquare representing 17% ,16% and 10%, respectively as illustrated in Figure 7. In addition, most of the users noted that they 'sometimes' use geosocial applications with almost a fifth of users 'always' using the location services. Foursquare users are more frequent users of the service than other services and 25% of the users have linked their accounts on different social networking applications. The questionnaire is divided up into four sections, was presented to participants in whole and takes roughly about 10 minutes to complete.

In what follows, the results from the different sections of the questionnaire are analysed.

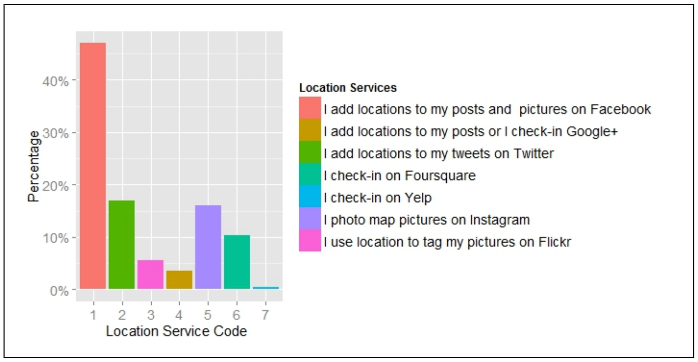


Figure 7: Percentage of the type of location services used by the 186 participants.

1) *Knowledge of Terms of Use and Privacy Policies for Social Networking Applications:* Here, the awareness of the terms of use and privacy policies are examined and analysed against users' profiles. In general, the majority of the users (81%) have not read terms of use or privacy policies of the social networking applications they use. Users were presented with the following typical statements representing the terms of use relating to location information and were asked to indicate whether they are aware of the information in the statements. Note that the following statements are representative of the terms of use of all the GeoSNs in question. The results are shown in Figure 8 grouped by the frequency of use.

- *Term 1: The application collects and stores your precise location (as a place name and/or a GPS point), even if you mark your location as private, for a possibly indefinite amount to time.*
- *Term 2: The application can use your location information in any way possible including sharing it with other applications or partners for various purposes (commercial or non-commercial).*
- *Term 3: If you share your location information, your friends and any other users are able to access and use it in any way possible.*
- *Term 4: The application can collect other personal information, such as your personal profile information and browsing history from other web applications.*

More than half (53%) of users acknowledged awareness of all of the statements and of those 73% have read the terms and policies. Most users (75%) are aware of statement 3, relating to the sharing of information with friends, but are generally unaware of statements 1 and 4, relating to how their location and other information may be collected and stored by the applications. It is interesting to note that frequent users of such application are generally unaware of such statements (49%) as demonstrated in Figure 8. Younger users aged between 15 and 34 tend to be more knowledgeable of these polices (60%), but gender does not seem to be a factor in these results.

2) *Perceptions of Possible Privacy Implications:* In this section, users' attitude towards the inference by the application of personal information is examined. In particular, the questions aim to gauge users' awareness of plausible inferences about their private places, activities at different times, their connections to other users, and possible knowledge of this information by the application. Participants were presented with 14 statements, shown below. They were then asked to indicate, for each statement, whether they are aware that the statement is possible and to score their reaction to the possibility of this statement as either 'OK', 'Uncomfortable' or 'Very Worried'. The first twelve statements refer to knowledge by the application itself, while the last two statements are reflection of the terms of use that suggest that the application can share the user's data with other users and third parties.

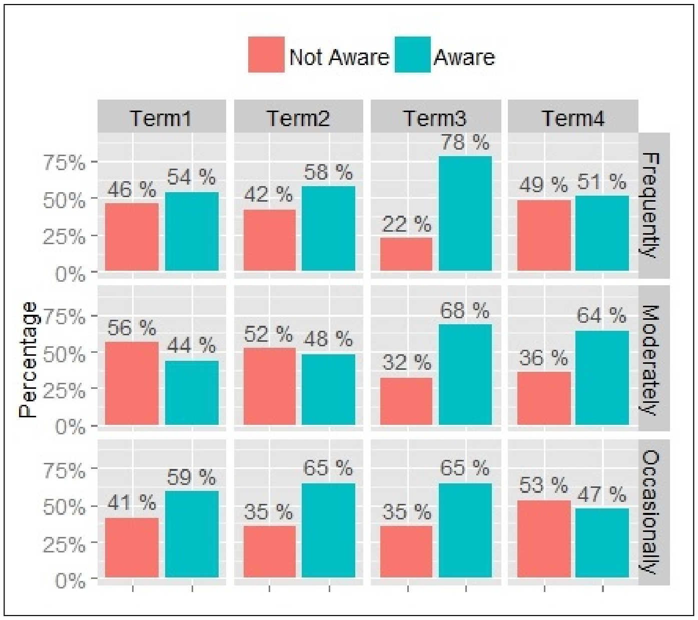


Figure 8: Users' awareness of general terms and policies of GeoSNs (Term1-Term4) grouped by the frequency of use.

- *S1: I can guess where your home is.*
- *S2: I can guess where your work place is.*
- *S3: I know which places you visit and at what times.*
- *S4: I can tell where you normally go and what you do in your weekends.*
- *S5: I can tell you where you go for lunch or what you do after work.*
- *S6: I know your favourite store (your favourite restaurant, your favourite coffee shop, etc.)*
- *S7: I can guess what you do when you are in a specific place.*
- *S8: I can guess when you are AWAY from home.*
- *S9: I can guess when you are OFF work.*
- *S10: I know who your friends are.*
- *S11: I know when and where you meet up with your friends.*
- *S12: I can guess which of your friends you see most.*
- *S13: Other people can know where you are at any point in time.*
- *S14: Other people can know what you are doing at any point in time.*

In terms of awareness, users seem to be most aware of statements S1, S2 and S10, regarding the location of home, place of work and friends, representing 88%, 89% and 93%, respectively. On the other hand, users are least aware of statements S5, S13 and S14 that relate to other users' knowledge of personal mobility patterns and activities, representing 34%, 37% and 40%. The awareness level of the users is demonstrated in Figure 9 grouped by the frequency of use.

Despite a reasonable level of awareness about the plausibility of these statement, users seemed to be relatively concerned about their privacy. 66% of users' reactions were either uncomfortable (41%) or 'very worried' (25%) as can be seen in Figure 10(a). Over half of the responses to S2 (awareness of workplace-53%) and S10 (awareness of friends-65%) were not concerned.

On the other hand, participants were most concerned with S13 and S14, with the 'Very Worried' category scoring 83% and 84%, respectively. S1 and S11, relating to the location of home and meetings with friends were rated most 'Uncomfortable' corresponding to 53% and 51%, respectively. Statement S8, suggesting the knowledge of user's absence from home and S13, indicating the possible knowledge of this information by other people presented a significant source of worry to users, with 45% and 42%, respectively indicating that they are 'Very Worried' about these statements.

It appears that users who read the terms and polices are more aware (by 9%) of the statements, while users who have not read the terms and polices were significantly 'Very Worried' (by 21%) than other users. Moreover, there is a positive correlation between the age of the participant and their level of awareness; level of awareness considerably increases with increase in age group, with the oldest active age group (35 to 44 years) scoring 89%. Yet, younger users, in the age group 15 to 34 years, tend to be relatively less concerned than older users (by 4%). The level of users' concern increases with the decrease in the frequency of use of the applications, where 76% of occasional users are concerned compared to 63% of frequent users. Users of Facebook and Instagram registered the highest degree of concern among all users of GeoSNs scoring 63% and 62%, respectively as shown in Figure 10(b). Again, gender does not seem to have any significant influence in this study.

3) Attitude to Privacy on Social Networks: The aim of this section of the questionnaire is to understand the users' reaction with regards to using the applications, given the knowledge of potential implications on privacy from the previous section.

61% of users stated that they would change the way they share their location information, 55% of whom are willing to stop sharing their location information completely, with the rest of the group indicating they would share it less often. Frequent users seem to be the most motivated to change their sharing behaviour (13% more than infrequent users), as illustrated in Figure 11, but they are also less willing to stop sharing the information and would prefer to share less frequently than the infrequent users (by 47%). Interestingly, users of location services are more tempted (by 10%) to change how they disclose their location information compared to users who have not used them. 57% of the first group of users want to share their location less frequently and 43% are willing to discontinue disclosing their location data. Younger users (15-34) are more willing to change their usage behaviour (by an average of 18%) and are even more willing to stop sharing location information completely (by an average of 10%) than older users. In this case, it seems that female users are more motivated to change their attitude regarding location disclosure (by 11%) than males, yet 60% of male participants suggested their willingness to discontinue using location services.

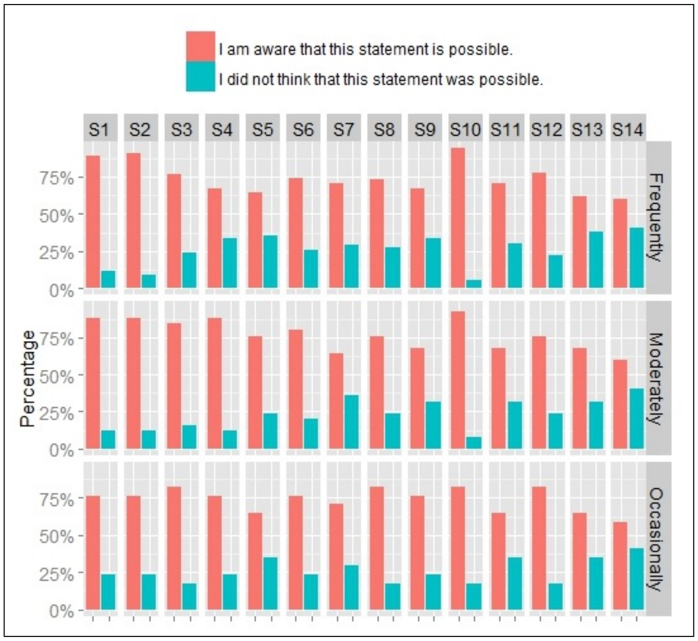
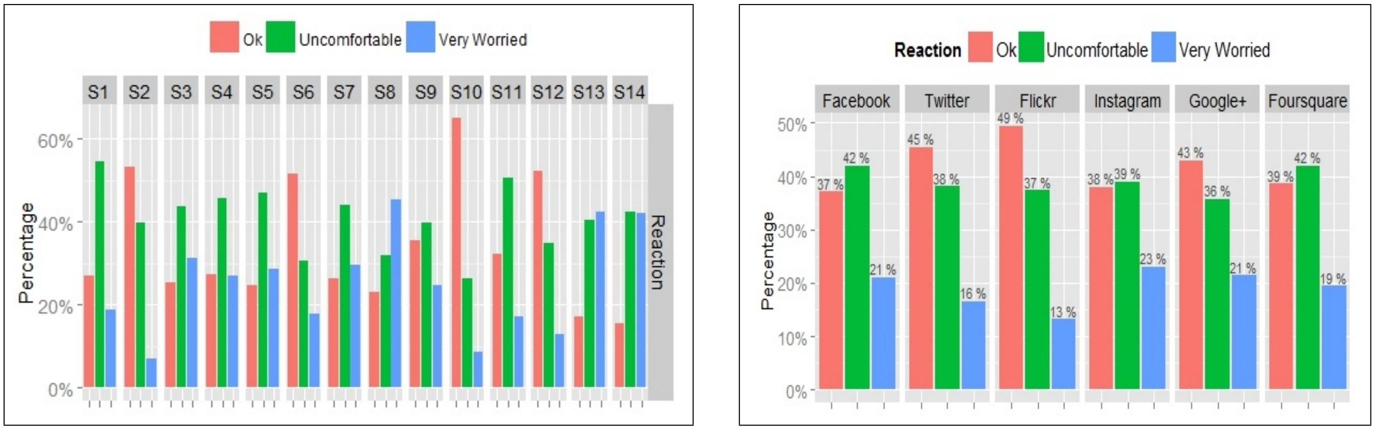


Figure 9: Users' awareness about potential information inferences (S1-S14) grouped by frequency of use of GeoSNs.

4) Managing Personal Information: In this section, users' views on managing and controlling access to their location information are explored. This includes several aspects related to what information is stored, how it is shared or viewed by the application and by others, and whether users need to manage access to their information. The following statements were presented to the participants who were asked to rate how often they would use them: 'All the time', 'Occasionally' or 'Never'.

- *C1: I would like to be able to turn off location sharing for specific durations of time.*
- *C2: I would like to turn off location sharing when I visit specific types of places.*
- *C3: I would like to decide how much of my location information history is stored and used by the application for example use only my check-in history for the last 7 days.*
- *C4: I would like to see the predicted personal information that the application stores about me based on my location information.*
- *C5: I would like to decide how people see my current location for example, exact place name, or a rough indication of where I am.*
- *C6: I would like to decide who can download my location information data.*



(a) (b)

Figure 10: (a) Users' reaction towards potential inferences grouped by the inference statements (S1-S14). (b) Data in (a) grouped by the GeoSNs used.

- *C7: I would like to know, and control, which information can be shared with other Web applications.*
- *C8: I would like to make my location information private seen only by myself and by the people I choose.*

Results are given in Figure 12(a) and show a significant desire to use these controls for location privacy. Overall, 76% of participants would like to apply those controls 'All the time', 20% are happy to apply them 'Occasionally', and only 4% of users will not consider these controls.

In general, C2, C6, C7 and C8 were most favoured controls, scoring over 97% each of users' responses. Controls C1, C6 and C7 were the most chosen controls to be applied all the time, representing 91%, 88% and 86% of users' responses, respectively. It is worth noting that users of different location services have similar acceptance rate for these control. Foursquare and Facebook users have the highest preference for applying the controls 'All the Time', corresponding to 76% and 75%, respectively.

A negative correlation appears to exist between users' tendency to use these privacy controls all the time and their age group. The youngest active age group of 15-24 years old has the highest desire for all-the-time application of controls representing 78% of this group's responses.

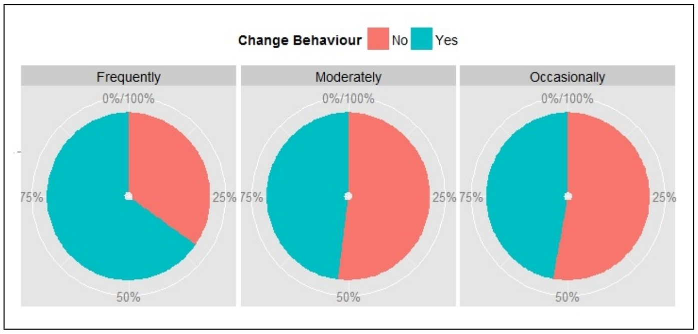


Figure 11: Users' attitude to location privacy risk, grouped by frequency of use of GeoSNs.

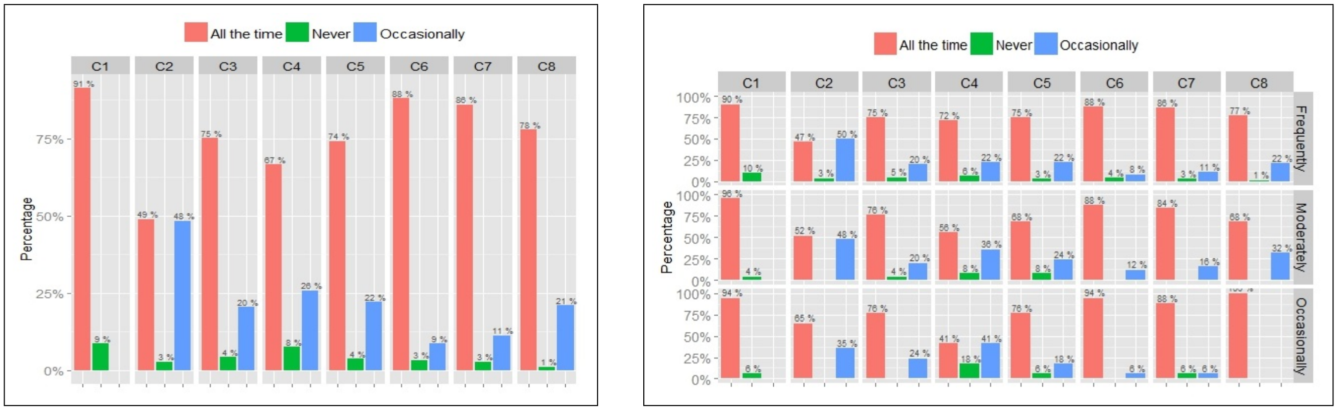
As expected, users who are tempted to change their location sharing behaviour have relatively higher motivation to use these controls representing 97% of this group's responses (4% higher than users who are reluctant to change). The factors of gender, whether users read the applications' terms or how frequent they use the social networks, as shown in Figure 12(b), seem to have minimal influence on their willingness to use these controls. In the future it will be useful to undertake a longitudinal study that tracks user behaviour over time to understand the factors that may influence their attitude to location privacy, for example the impact of friends and age group.

VI. Discussions and Conclusions

The proliferation of location-based GeoSNs and the large-scale uptake by users suggest the urgency and importance of studying privacy implications of personal information collected by these networks. Identifying user profiles is a goal of many businesses that is now commonly accepted by users for the purpose of improving the quality of service. However, GeoSNs do not explicitly present similar business goals and thus their motivations for collecting and sharing personal location information are not clear. Also, the issue is complicated as the data collected may be shared or accessed by other users and applications. The results of this study highlight the possible implications to user privacy and the need for developing means for raising the user awareness of these issues, and possibly also giving the user control on managing access to their data.

The data analysis experiment conducted here shows the amount and types of personal information that can be inferred using location data. Users' spatiotemporal mobility tracks can be analysed to identify where they are, where they are likely to be, and sometimes more significantly, where they are not present. Tracking user location data may also give indications to their preferred activities, places, habits and friendship community.

As can be expected, the more frequent the applications are used, the more dense the spatiotemporal history of user data collected and the more certainty in the derived information extracted from this data. Whilst the statistical analysis carried



(a)

(b)

Figure 12: (a) Users' desire to use location privacy controls grouped by statement of controls C1-C8 (see section V-B4). (b) Data in (a) grouped by frequency of use of GeoSNs.

out in this study highlight some of the basic and interesting inferences that can be made, more sophisticated location-based inference methods can be developed to infer, for example, the probability of future movements, methods of transport and places visited. The now common practice of linking user accounts in several GeoSNs increases the availability of data and compounds the privacy risks to users, who sign up to different, possibly contradicting, terms of use and policies of different applications. For example, developers now use the Twitter API to collect user check-ins in Foursquare.

The questionnaire conducted in Section V provides valuable insights that convey many aspects of location privacy on the Social Web from the perspective of the end user. The main and (possibly only) means of communicating how the collected user information may be used and exploited by the application is described in the application's terms of use. It is clear from the results of the questionnaire undertaken that the majority of users, especially those who use location services, do not read the terms of use and policy documents. The findings also indicate that users are aware of the potential information, and possible derivatives thereof, stored by the application. However, it appears that they are also quite concerned about the privacy implications. This apparently contradicting findings may be due to that such awareness and concerns are evident when users are actively questioned about these issues, but are somewhat screened from the users' minds during the continuous use of the application. The study also suggests that users may not fully understand the privacy implications, where their level of concern was much more pronounced when faced with statements that indicate that other people may be aware of their location information in comparison to statements indicating that the application holds such information.

The study reveals that there is a strong need for the users to be continuously aware of their data, how it is stored and to have the ability to control access to and visibility of their location data sets. Further research is needed into methods that enhance the communication of the information by the applications as well as method to allow users to better understand and control their personal profiles on such networks.

References

[1] F. Alrayes and A. Abdelmoty, "Privacy concerns in location-based social networks," in GEOProcessing 2014: The Sixth International Conference on Advanced Geographic Information Systems, Applications, and Services. IARIA, 2014, pp. 105–114.

[2] C. Vicente, D. Freni, C. Bettini, and C. Jensen, "Location-related privacy in geo-social networks," IEEE Internet Computing, vol. 15, no. 3, 2011, pp. 20–27.

[3] D. Riboni, L. Pareschi, and C. Bettini, "Privacy in georeferenced context-aware services: a survey," in Privacy in Location-Based Applications. Springer Verlag, 2009, pp. 151–172.

[4] S. Gambs, O. Heen, and C. Potin, "A comparative privacy analysis of geosocial networks," in SPRINGL '11 Proceedings of the 4th ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS, 2011, pp. 33–40.

[5] R. Shokri, G. Theodorakopoulos, J.-Y. Le Boudec, and J.-P. Hubaux, "Quantifying location privacy," in IEEE Symposium on Security and Privacy, 2011, pp. 247–262.

[6] J. Tsai, P. Kelley, P. Drielsma, L. Cranor, J. Hong, and N. Sadeh, "Who's viewed you? the impact of feedback in a mobile location-sharing application," in CHI '09, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2009, pp. 2003–2012.

[7] N. Sadeh, J. Hong, L. Cranor, I. Fette, P. Kelley, M. Prabaker, and J. Rao, "Understanding and capturing peoples privacy policies in a mobile social networking application," Personal and Ubiquitous Computing, vol. 13, no. 6, 2008, pp. 401–412.

[8] H. Gao, J. Hu, T. Huang, J. Wang, and Y. Chen, "Security issues in online social networks," Internet Computing, IEEE, vol. 15, no. 4, 2011, pp. 56–63.

[9] J. Nagy and P. Pecho, "Social networks security," in Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on. IEEE, 2009, pp. 321–325.

[10] P. Joshi and C.-C. Kuo, "Security and privacy in online social networks: A survey," in Multimedia and Expo (ICME), 2011 IEEE International Conference on. IEEE, 2011, pp. 1–6.

[11] S. Patil and J. Lai, "Who gets to know what when: configuring privacy permissions in an awareness application," in Proceedings of the SIGCHI conference on human factors in computing systems (CHI 2005), 2005, pp. 101–110.

[12] P. Kelley, M. Benisch, L. Cranor, and N. Sadeh, "When are users comfortable sharing locations with advertisers?" in Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11, 2011, pp. 2449–2452.

[13] A. Brush, J. Krumm, and J. Scott, "Exploring end user preferences for location obfuscation, location-based services, and the value of location," in Proceedings of the 12th ACM international conference on Ubiquitous computing - Ubicomp '10, 2010, pp. 95–104.

[14] K. Tang, J. Hong, and D. Siewiorek, “Understanding how visual representations of location feeds affect end-user privacy concerns,” in Proceedings of the 13th international conference on Ubiquitous computing - UbiComp ’11, 2011, pp. 207–216.

[15] J. Lindqvist, J. Cranshaw, J. Wiese, J. Hong, and J. Zimmerman, “I’m the mayor of my house: examining why people use foursquare-a social-driven location sharing application,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI ’11, 2011, pp. 2409–2418.

[16] L. Jin, X. Long, and J. Joshi, “Towards understanding residential privacy by analyzing users’ activities in Foursquare,” in Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security - BADGERS ’12, 2012, pp. 25–32.

[17] Z. Cheng, J. Caverlee, and K. Lee, “You are where you tweet: a content-based approach to geo-locating Twitter users,” in Proceedings of the 19th ACM international conference on Information and Knowledge Management CIKM ’10, 2010, pp. 759–768.

[18] T. Pontes, M. Vasconcelos, J. Almeida, P. Kumaraguru, and V. Almeida, “We know where you live?: privacy characterization of foursquare behavior,” in UbiComp ’12 Proceedings of the 2012 ACM Conference on Ubiquitous Computing, 2012, pp. 898–905.

[19] A. Sadilek, H. Kautz, and J. Bigham, “Finding your friends and following them to where you are,” in Proceedings of the fifth ACM international conference on Web Search and Data Mining, WSDM ’12, 2012, pp. 723–732.

[20] H. Gao, J. Tang, and H. Liu, “Exploring social-historical ties on location-based social networks,” in Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media, 2012, pp. 1140–121.

[21] ——, “gSCorr: modeling geo-social correlations for new check-ins on location-based social networks,” in Proceedings of the 21st ACM international Conference on Information and Knowledge Management, CIKM ’12, 2012, pp. 1582–1586.

[22] D. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, and J. Kleinberg, “Inferring social ties from geographic coincidences,” in Proceedings of the National Academy of Sciences of the United States of America, vol. 107, no. 52, 2010, pp. 22 436–22 441.

[23] S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo, “Socio-spatial properties of online location-based social networks,” in ICWSM, 2011, pp. 329–336.

[24] S. Scellato, A. Noulas, and C. Mascolo, “Exploiting place features in link prediction on location-based social networks categories and subject descriptors,” in Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, 2011, pp. 1046–1054.

[25] D. Dearman and K. Truong, “Identifying the activities supported by locations with community-authored content,” in Proceedings of the 12th ACM international conference on Ubiquitous computing, 2010, pp. 23–32.

[26] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil, “An empirical study of geographic user activity patterns in foursquare,” in ICWSM, 2011, pp. 70–73.

[27] Z. Cheng, J. Caverlee, K. Lee, and D. Sui, “Exploring millions of footprints in location sharing services,” in ICWSM, vol. 2010, 2011, pp. 81–88.

[28] D. Preotiuc-Pietro and T. Cohn, “Mining user behaviours: a study of check-in patterns in location based social networks,” Web Science, 2013.

[29] “Instgram location endpoints,” Nov. 2014. [Online]. Available: http://instagram.com/developer/endpoints/locations/#

[30] A. Popescu, G. Grefenstette, and P.-A. Moëllic, “Gazetiki: automatic creation of a geographical gazetteer,” in ACM/IEEE Joint Conference on Digital Libraries. ACM, 2008, pp. 85–93.

[31] “Foursqure terms of use,” Nov. 2014. [Online]. Available: https://foursquare.com/legal/terms

[32] “Twitter terms of use,” Nov. 2014. [Online]. Available: https://twitter.com/tos

[33] “R project,” Nov. 2014. [Online]. Available: http://www.r-project.org

[34] “World domination summit,” Nov. 2014. [Online]. Available: http://worlddominationsummit.com/faq/#primary-content

A Generalized View on Pseudonyms and Domain Specific Local Identifiers

Lessons Learned from Various Use Cases

Uwe Roth
SANTEC
CRP Henri Tudor
L-1855 Luxembourg, Luxembourg
uwe.roth@tudor.lu

***Abstract*—Pseudonymisation as a data privacy concept for medical data is not new. The process of pseudonymisation gets difficult in concrete use-case setups and the different variations of data flow between those who collect, who store, and who access the data. In all cases, questions have to be answered about, who has access to the demographics of a person, who has access to the pseudonym, and finally, who creates the pseudonym. Since a fundamental part of the pseudonym creation depends on the identification of a person on base of its demographics, things even get more difficult in case of unclear matching decisions, management of wrong matching or update of demographic information. In this journal article, a unified view on pseudonyms is proposed. Pseudonyms are treated as a local identifier in an identifier domain, but in a domain that has no demographics. Additionally, persistent identifiers are introduced that allow the handling of updates and internal matching reconsiderations. Finally, two concepts for pseudonymisation are shown: First, a National Pseudonymisation Service is sketched with focus on resistance against update problems and wrong matching decisions. It is designed to cover every possible variation of the exchange of local identifiers between a source of personal data and the storage destination. Second, an algorithm for the pseudonym creation from a person identifier is described. This algorithm is needed if the pseudonymisation is not performed by an external service but in-house and in case of limited number space of the pseudonyms. Both solutions are suitable to solve a huge variety of pseudonymisation setups, as it is demanded by researchers of clinical trials and studies.**

Keywords-patient privacy-enhancing technologies; secure patient data storage; pseudonymisation; local identifer; identifier domain.

I. INTRODUCTION

This article is an extended version of [1], which covers the algorithm for the generation of pseudonyms with a limited number of bits.

Pseudonymisation is a process where demographics and identifier of a person are removed out of an information record and replaced by a pseudonym. This step is demanded to protect the privacy of patients in cases of secondary usage of medical data, e.g., for research or statistical purposes. In these cases knowledge about the identity of the person is unnecessary and therefore must be protected against disclosure. In contrast to anonymisation, a pseudonym allows to link data from several sources to the same person, which helps to improve the quality of the research or statistics.

An example for the need of pseudonymization is the storage of medical data, samples, blood, and urine in biobanks. Researchers are not interested in the identity of the person behind this material. A pseudonym is needed to link all samples that have been taken from the same person at different locations and during different collection events. The pseudonym will not only allow the linkage to the same person but also allows protecting the identity of the patient behind the sensitive data.

One part of this article describes a generalized concept on how identities of patients and their pseudonyms are used and managed (including identity matching, linkage of identifiers from different domains) to securely exchange data. Despite the fact that these problems are discussed in many publications (e.g., [2] and [3]) this article gives a generalized overview of how a source-destination relation can be defined.

The main idea behind the generalization is the concept of local identifiers of identifier domains that are either bound to demographics or not. With the generalization of pseudonyms as local identifiers in a domain without demographics, transitions of identifiers between certain identifier domains become only a matter of permissions, e.g., permission to pseudonymise, permission to re-identify. So the main cases that are discussed in the article differentiate the variations of visibility of demographics, local identifiers and pseudonyms amongst the source of data and the destination storage.

All cases can be implemented by the use of a pseudonymisation service as a trusted third party. The article defines the fundamental services of the pseudonymisation service that are needed to treat all identified cases. They have been specified for the National Pseudonymisation Service of Luxembourg, which is solely responsible for the management of persons and the transition of the identifiers between the different identifier domains. The National Pseudonymisation Service will not perform pseudonymisation on medical data, nor will it have access to medical data.

With the provisioning of demographics in a certain domain (e.g., hospital, laboratory), the introduction of faulty data is likely. The update of such data might lead to a revised decision at the National Pseudonymisation Service, i.e., demographics from a certain domain now match a different known person or it is assumed that the persons is unknown yet. This has consequences at the destination side and requires an update of the pseudonym for some of the stored data. With the introduction of persistent identifiers that are linked to the initial matching decision, update of only the pseudonyms that are concerned is possible.

Central or national pseudonymisation services run as Trusted Third Parties for example in the Netherlands (ZorgTTP [4]), and in Germany the Patient Identifier (PID) generator in combination with a pseudonymisation service of TMF (Telematikplattform für Medizinische Forschungsnetze e.V.) is well known [5]. These solutions mainly provide global person identifiers for identified persons, which can be used to create domain specific pseudonyms. Mechanisms and information to handle faulty matching decisions after the update of demographics are not foreseen.

In the TMF solution, the visibility of the demographics and the pseudonym at source and destination are restricted by passing the (encrypted) medical data, together with the global identifier (from the PID generator) through the pseudonymisation service. Such a setup on national level would require, that the National Pseudonymisation Service must be able to access services in the research domains to push the pseudonymized data to it. As a consequence, researchers need to maintain a service in their Demilitarized Zone (DMZ) that is able to receive the pseudonymized data. In the proposed solution, the pseudonymisation service acts only as a passive service that can be accessed from Intranets without the need of a DMZ. Additionally the solution does not need to bypass medical data and therefore is able to manage more requests per time.

An alternative to the use of a National Pseudonymisation Service is the implementation of a local in-house pseudonymization, which means that the pseudonym is calculated either at the data source or the storage destination out of a given person identifier without the use of an external service. In such a setup no matching decisions will take place and a person requires a stable person identifier.

In both cases (National Pseudonymisation Service or in-house pseudonymisation) the pseudonym number itself has to be calculated or determined at one point in time. There are several options to create a pseudonym with a given set of demographics. Some of these techniques base on hashing or encryption of a unique identifying number of the person. Others simply chose a random number and link this number with the identity.

Current hashing and encryption algorithms work with 128 bits minimum, which might be too much in some cases, e.g., the pseudonym must be 31 bit unsigned integer. In that case, the outcome of the process must be cropped to the desired bit-length, which leads to an unpredictable risk for pseudonym collisions.

Research that takes smaller number of bits into account is known as small-domain pseudo random permutation or small-domain cipher (e.g., [6][7][8]). Solutions that base on this research use techniques that are also used in symmetric encryption (e.g., Advanced Encryption Standard AES [9]) or hashing algorithms (e.g., Secure Hash Algorithm SHA [10]): Permutation, rotation, transformation, and diffusion of the given bit-set of data. A similar research area that uses the same tools deals is Format Preserving Encryption (FPE) (e.g., [11]). Here, more focus is made on the format of the encrypted block of data, which also includes the format on char- or word-level. The FALDUM Code [12] as another example tries to create a code with error correction properties and good readability.

For all proposals, it is difficult to estimate how secure these algorithms finally are and how difficult it is to re-compute the person identifier with a given pseudonym. Cryptanalysis on existing symmetric encryption algorithms and hashing algorithms have shown, that weaknesses can be found years after the algorithms has been proposed (e.g., [13]).

Therefore, an alternative pseudonym calculation algorithm is proposed to calculate pseudonyms from a person identifier on the base of a chosen primitive root of a fixed prime number. This calculation is more similar to asymmetric encryption techniques (e.g., the RSA algorithm [14]) or the Diffie-Hellman-Key Exchange protocol [15].

The algorithm guarantees a collision free pseudo-random distribution of the pseudonyms. The pseudonymisation algorithm acts as a one-way function if all of the calculation parameters are kept secret.

The article is structured as follows:

In *Section II – Methods*, the concept of identifier of persons and identifier domains and its relation to pseudonyms is introduced. Later, the main cases of data transmission between a source system and a destination storage are listed, including the different visibilities of local identifiers at source and destination. Persistent identifiers are introduced to solve two problematic cases that might get relevant in case of update of demographics. Then a look at the number space of identifiers and the existence of demographics in a certain identifier domain is taken. In a setup of a National Pseudonymisation Service, properties and permissions of systems and domains have to be defined. Then finally, the main identity related services of the National Pseudonymisation Service will be outlined. Since the National Pseudonymisation Service use an existing Master Patient Index for matching decisions, aspects of this relation will be discussed. The section ends with discussions about the creation of new local identifiers, especially the calculation of local identifier with small number of bits.

In *Section III – Results*, the use of the National Pseudonymisation Service and the use of an in-house pseudonymisation solution will be shown on existing use cases that have been implemented already or which are in planning.

The paper ends with *Section IV – Conclusion and Future Work*, in which the positive effects of the proposed solutions for researchers will be outlined.

II. Methods

The generalized concept of a National Pseudonymisation Service (NPS) and of an in-house pseudonymisation solution bases on use cases that have been identifier by questioning various researchers in the field of clinical and population

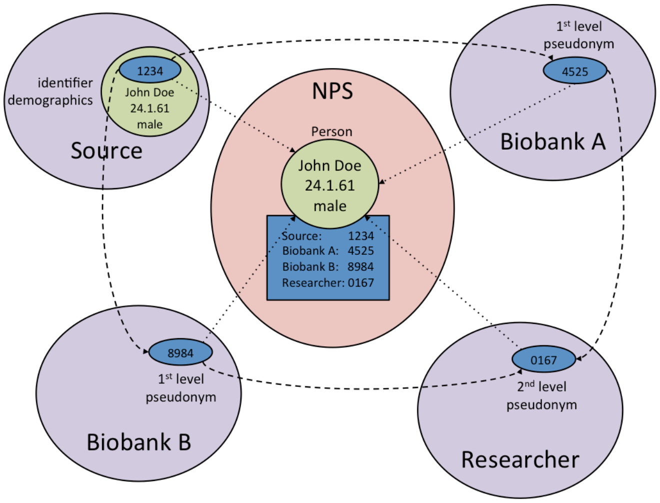


Figure 1. Identities and Domains

based studies. First, some terms must be clarified; later, these cases will be discussed.

A. *Identifier of persons and identifier domains*

In the digital world, the use of identifier of persons is quite common. It simplifies the linkage of data of the same persons, if unique identifiers are used. This linkage is quite complicated if only demographics (e.g., name, address, birthday) are given.

1) *Local identifier and identifier domain*

The concept of (local) identifiers for persons that are only valid in a certain local context is one of the basic concepts of the IHE Patient Identifier Cross Referencing (PIX) Integration Profile [16][17], as it is implemented inside hospitals or laboratories. Usually different systems (e.g., storage systems, imaging systems) use different identifiers inside the same institution to identify the same person. The Patient Identifier Cross-reference Manager enables the systems to communicate with each other, even if they use different identifier for the same person. This is solved by so called identifier domains for the different systems. Usually, the same person should only have one identifier inside an identifier domain. This concept cannot only be used for the exchange of data inside an institution but also between different institutions (different domains), for which a person has different patient identifiers (local identifier).

The local identifier of a person in one domain is different from the local identifier of the same person in another domain. Without help of the Patient Identifier Cross-reference Manager it is difficult to translate the link of persons between the two domains.

The concept of local identifier and identifier domains is used in the concept of the National Pseudonymisation Service. Identifier domains not only describe institutions but also might identify applications or application contexts, e.g., national laboratory-application, clinical study about cancer. The identifier domain usually is identified by a unique OID (Object Identifier) [18].

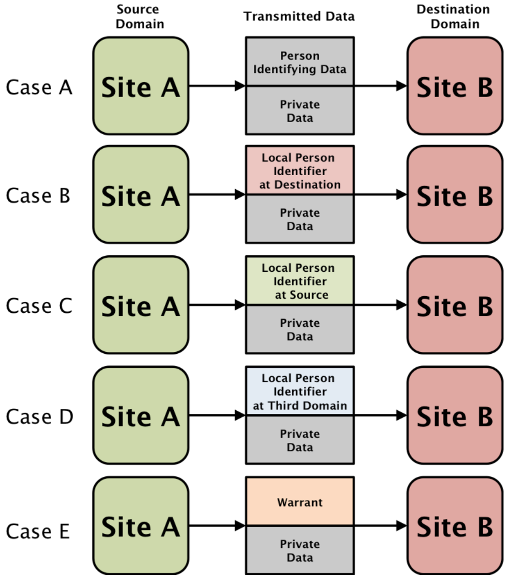


Figure 2. Different cases of transmitting data

2) *Pseudonym*

In the proposed concept, a pseudonym is seen as a local identifier inside an identifier domain where no demographics are available.

Pseudonyms from different domains must be different. Having a local identifier from one domain must not allow calculating the pseudonym from another domain, except the domain is responsible for the creation of the pseudonym. This statement ensures that it is not possible to break the pseudonymisation on known identifiers.

3) *2nd-level pseudonym*

As for pseudonyms, a 2nd-level pseudonym is also only a local identifier in a certain identifier domain where no demographics are available. In this case, the source of data is a domain that identifies persons by pseudonyms and not by demographics.

2nd-level pseudonyms in an identifier domain can be linked to the same person, even if the 1st-level pseudonym was from different domains.

Example (Figure 1): Medical data of a person are sent to Biobank A that works with 1st-level pseudonyms. Medical data of the same person are sent to Biobank B that works with different 1st-level pseudonyms. Data of both biobanks are sent to a researcher who works with 2nd-level pseudonyms. The researcher is able to link data of both biobanks to the same person, in case of the same 2nd-level pseudonym.

It is clear that such a scenario in reality requires approval by ethics commissions or data protection authorities.

B. *Main cases of data transmission*

After being familiar with the terms *local identifier* and *identifier domains*, it is possible to describe the main cases of transmitting data between a source and a destination. The described use cases cover cases that include the use of a

National Pseudonymisation Service and the use of an in-house pseudonymisation, with a stronger focus on the design of the National Pseudonymisation Service.

In the proposed setup the communication between source and destination systems is direct, so no system is involved during the transmission of medical data between source and destination that modifies the transmitted data. This is not only true for the in-house pseudonymisation but also in case of the use of the National Pseudonymisation Service. The National Pseudonymisation Service is defined solely as a passive service that is used to identify persons and request or to manage local identifies. It will not allow the bypass of medical data from source to destination. Also, it will not perform a pseudonymisation of medical data on the fly (i.e., replace demographics in the medical data by pseudonyms).

The question that results from these restrictions is: What information is send from source to destination (apart from the medical data) that allows the mapping of the medical data to a certain person at the destination?

There are several options to answer this question. The five possible cases that describe these options are shown in Figure 2:

A. Demographics of the person are exchanged.
B. The local identifier from destination domain is exchanged.
C. The local identifier from the source domain is exchanged.
D. The local identifier form a third domain is exchanged.
E. A warrant is exchanged that can be used by the destination to request the local identifier from its domain by showing the warrant.

All cases do not make an assumption on how the data and information is transmitted between source and destination. It does not have to be electronically only. Alternatively, this data could be send by the use of physical objects (e.g., as barcode on paper or box).

1) Case A: Demographics of the person are exchanged

In this case private data of a person is exchanged between source and destination together with the demographics of the person. So the destination is forced to link data from the same identity on base of the given demographics. This could be done by the use of a local Mater Patient Index (MPI) or by the use of a National Pseudonymisation Service. Anyway, it is clear that this local identifier is not a pseudonym, as the identity of the person is known.

2) Case B: The local identifier from the destination domain is exchanged

In this case private data of a person is exchanged between source and destination together with the local identifier of the person of the destination domain attached to it. So the sources need to calculate, determinate or know the local identifier of the destination domain on base of its own local identifier or the known demographics of the given person. Alternatively, it needs to ask the National Pseudonymisation Service to provide this identifier of the destination.

As a consequence, all source systems from all source domains will know the local identifiers or pseudonyms from the destination domain but not vice versa. In case of the use of a National Pseudonymisation Service, the sources systems needs permissions to request the local identifier of the destination domain on base of its own local identifier or demographics.

3) Case C: The local identifier from source domain is exchanged

In this case private data of a person is exchanged between source and destination together with the local identifier of the person at the source domain attached to it.

As a consequence the destination system of the destination domain will know the local identifiers from the source domain but not vice versa, so the local identifier or pseudonym that is used at the destination is hidden to all sources.

In case of in-house pseudonymisation, this case only makes sense in case of one source only, otherwise it will be impossible to link identifier from different sources to the same person. This limitation does not exist in a setup with the use of a National Pseudonymisation Service, for which the destination needs permission to translate the local identifiers of the sources to its local domain identifier.

4) Case D: The local identifier from a third domain is exchanged

This case introduces a third identifier domain. This case makes sense if such a third domain is created especially for the exchange between source and destination and nowhere else. In such a setup local identifiers from a source will not be disclosed at the destination and vice versa. Source and destination systems must only use the identifier of the third domain during the exchange of the private data and not for the storage of the private data.

This case allows different variations by using in-house pseudonymisation or the National Pseudonymisation Service during the transition of the identifier between source to the third domain, and between third to the destination domain. As for Case C, an in-house pseudonymisation between sources and third domain is only useful in case of only one source domain, because it is impossible to define a calculation or determination process that would allow the transition of local identifiers of the same person from different sources that result in the same identifier in the third domain.

The translation between the identifier of the third domain and destination domain can be performed in-house or at the National Pseudonymisation Service.

5) Case E: A warrant is exchanged

In this use case private data of a person is exchanged between source and destination together with a warrant attached to it. This use case requires the use of the National Pseudonymisation Service and does not work with in-house pseudonymisation.

The warrant is created and/or managed by the National Pseudonymisation Service on base on information, provided by the source (e.g., local identifier or demographics). The

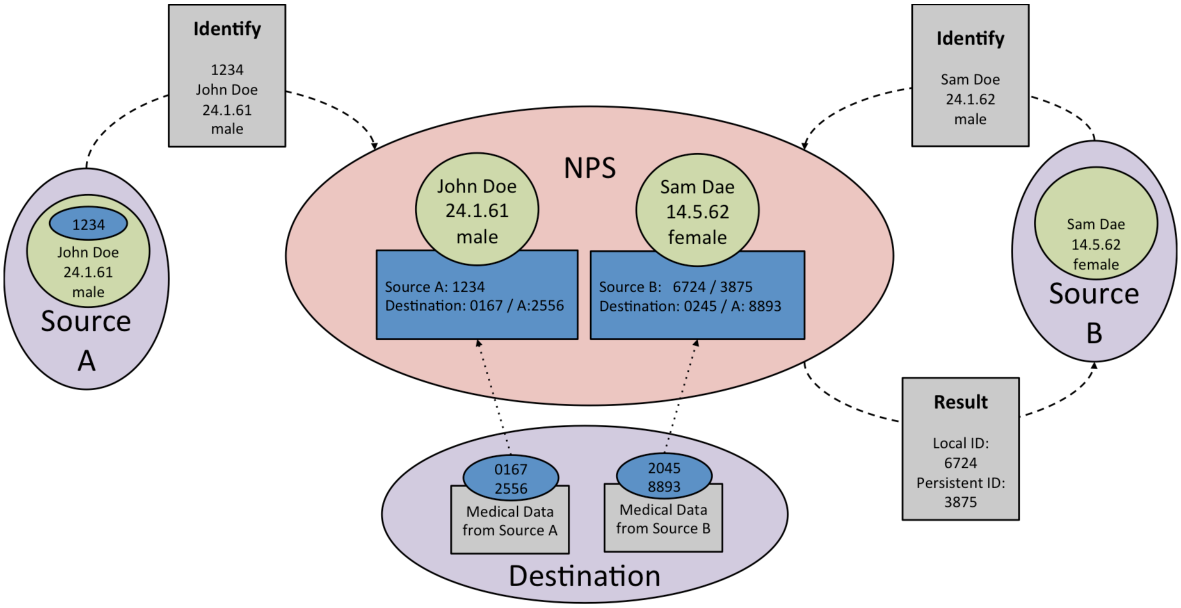


Figure 3. Persistent identifier and initial identification

destination will then be able to retrieve the identifier belonging to the destination domain on base of the warrant.

In this case, the source does not know the local identifiers or pseudonyms in the destination domain and the destination does not know the demographics at the source domain.

In contrast to Case D and the use of identifiers from a third domain, the warrant can be managed by the source and might be defined with at time-to-live. The warrant should not be used as a replacement of a local identifier because they are not unique in case of the same person. Additionally, the National Pseudonymisation Service might delete the warrant out of its systems after use.

The warrant-based approach might be used in cases of re-identification of patients. In that case, a warrant is requested by the destination on base of the pseudonym and the source is able to re-identify the patient on base of the warrant.

C. *Identifier management*

Usually, hospital information systems or equivalent systems manage local identifier for patients themselves. In this case, the local identifier is created inside the identifier domain of the data sources. The identifier domain guarantees that the person behind the local identifier never changes, even if the demographics of that person change significant. This is an important requirement. In the future, it might be possible that two identifiers are merged because they have been identified as doublets of the same person. But an identifier never changes the link to the individual person.

Not all identifier domains manage identifiers themselves. As an example, collection sites of a clinical study might be located at hospitals, but have no access to the hospital information systems and therefore not to the local identifier of that hospital. In that case a new local identifier has to be created for the collection site domain. The National Pseudonymisation Service can overtake this task on the base of given demographics.

In the National Pseudonymisation Service, it must be configured for each data source, if it creates and manages identifiers themself or if the National Pseudonymisation Service has to take responsibility for this.

D. *Persistent local identifier*

The National Pseudonymisation Service decides with given demographics, if the demographics match with the demographics of a known person or not. If demographics of a person are updated at a source, this might lead to a different matching decision at the National Pseudonymisation Service, so the demographics are linked to a different person.

Sources who manage local identifiers in their domain are not affected by this decision because the local identifier of the person at the source will not change. For sources and destinations with local identifiers management by the National Pseudonymisation Service, things are different: some data sets with an associated local identifier might need to be changed in a way that reflects the new matching decision, i.e., local identifiers of some datasets need to be updated too. The identification of these datasets on base of the current local identifier is not sufficient, because for some datasets from different sources, the change must not be performed.

To solve this issue, an additional persistent local identifier is introduced. The persistent local identifier will never change, regardless of updates of demographics. It can be used to provide update information for exactly those entities that are affected by the update decision.

The persistent identifier is an addition to the local identifier inside an identifier domain and is linked to the demographics that were used during the first identification step of the demographics from a source at the National Pseudonymisation Service.

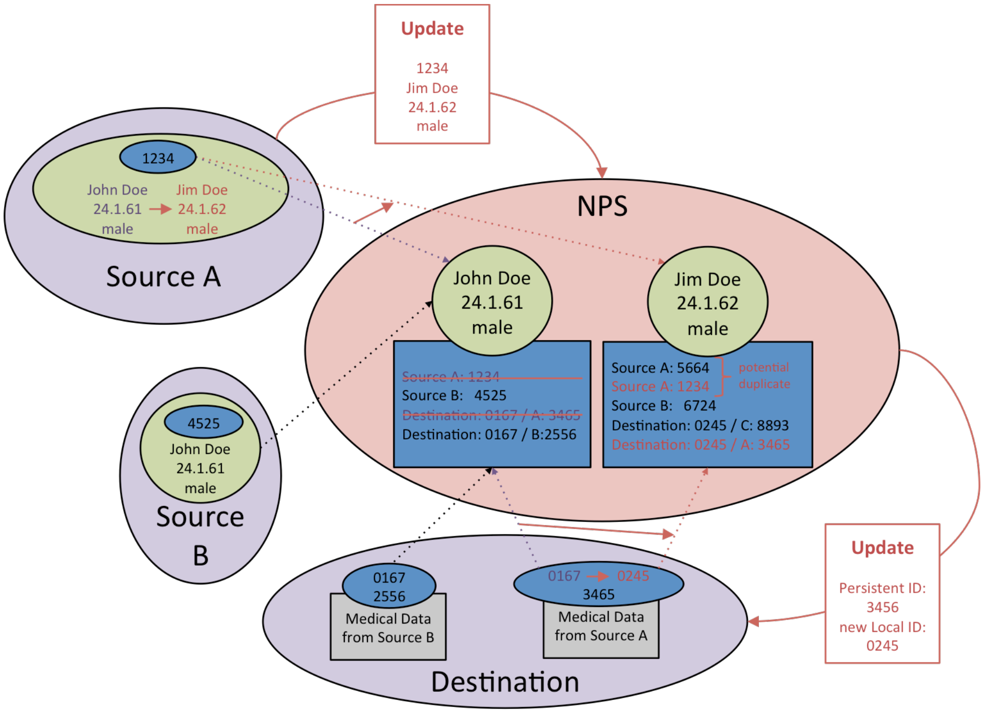


Figure 4. Problematic case: Update affects destination

At one point in time, a data source needs to identify demographics of a person at the National Pseudonymisation Service, either to make it aware of the local identifier in the source domain or to request a local identifier on base of the demographics. The persistent identifier is bound to that identification process.

In the example of Figure 3, Source A is identifying John Doe together with its local identifier 1234. Medical data that is sent from Source A to the destination is linked to that person at the destination via the local identifier 0167 and the persistent identifier 2556. In the same example, Source B identifies demographics of Sam Dae without a locally managed local identifier. This source will receive the local identifier 6724 from the National Pseudonymisation Service plus a persistent identifier 3865. If Source B identifies a person with the same demographics in future, it will receive the same local identifier 6724 but always with a different persistent identifier.

E. Problematic cases

The persistent identifier can be used to solve two problematic cases:

- Update affects destination
- Update affects source

1) Update affects destination

Two sources from different domains (Example Figure 4: Source A, B) provide demographics that lead to the same local identifier/pseudonym at the destination (0167). Then one of the sources (Source A) updates the demographics and the National Pseudonymisation Service decides that the previous matching decision was wrong and that this new demographics belongs to a different person. So the local identifier (1234 of Source A) is re-linked in the National Pseudonymisation Service to a different or new person. On base of the persistent identifier (3465), the destination can be informed to update the local identifier (0245). This affects only the medical data that has its origin in Source A.

One could argue that a persistent identifier could be avoided, if the destination would store information about the source domain together with the local identifier. In the example an update then would be: Update data from Source A with local identifier 0167 to the new local identifier 0245.

This argument is true, but there are good data protection arguments to hide the origin of the data at the destination. The persistent identifier in that case acts as a pseudonymisation of the source.

2) Update affects source

The National Pseudonymisation Service manages the local identifiers of a source domain (Example Figure 5: Source A), so the National Pseudonymisation Service provides the local identifiers plus a persistent identifier after identification of demographics.

During two independent events, demographics are identified at the National Pseudonymisation Service that lead to the same local identifier at the source (1234) but with different additional persistent identifiers (2347, 5678). Then later, one set of demographics (identified by local identifier plus persistent identifier: 1234 / 2347) is updated and the National Pseudonymisation Service decides that the demographics belong to a different person as previously suggested (Figure 6). So the data that was provided in one event at Source A has the wrong local identifier and needs to be changed to the new local identifier (5667). This change does not affect the local identifier from the second event.

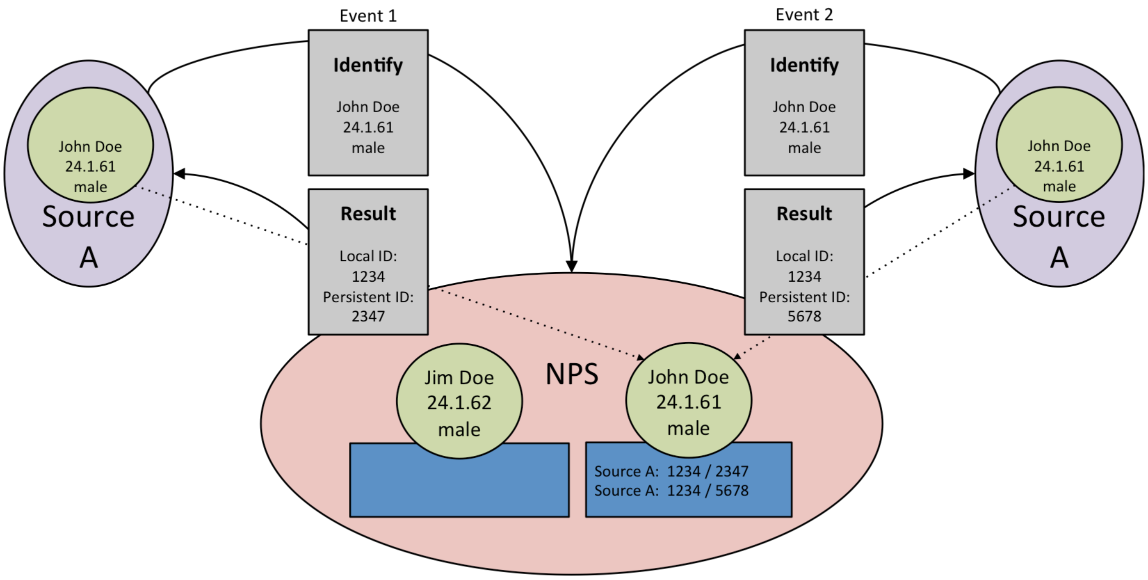


Figure 5. Problematic case: Update affects source (initial state)

One might argue that such a use case is not likely, especially in case of information systems inside the source domain. In case of clinical studies, sometimes the management of identities depends on papers, Excel sheets or other unreliable tools. So it is not an unrealistic scenario that nurses collect samples at different collection events and use the National Pseudonymisation Service to retrieve a local identifier on base of re-typed erroneous demographics, which later needs to be updated.

F. Avoidance of persistent local identifiers

In the case of local identifiers of a source domain that is managed by the National Pseudonymisation Service, the use of persistent local identifiers is one way to manage updates. An alternative approach can avoid the use of persistent local identifiers. It foresees that each matching request at the National Pseudonymisation Service that is performed without the use of a local identifier will lead to a new local identifier, regardless whether the demographics match a known person or not.

As for local identifiers that are managed by the sources, this identifier will never change even after update of demographics. The National Pseudonymisation Service can be asked, for which local identifiers it assumes that they belong to the same identity. This list might change after demographics are updated for a given local identifier.

So there are two options to treat potential update problems at domains that do not create or manage local identifiers: Either a persistent identifier is provided together with the local identifier, or always new local identifier are created even if the National Pseudonymisation Service assumes that the demographics belong to a known person.

G. Identifier domain and identifier number space

In case of local identifier created and managed by the National Pseudonymisation Service, it is suggested, that the number is a purely (pseudo-)random integer number from the range zero to a maximal number. It does not include any information that is linked to the demographics of the person. The maximum has to be defined per identifier domain at the National Pseudonymisation Service.

It is up to the users of the local identifier if they encode the number into a character representation or if they add error correction or error detection codes, e.g., to make it human readable. During communication with the National Pseudonymisation Service, only the integer representation must be used.

H. Identifier domain and availability of demographics

One can distinguish between domains where demographics are available and domains where demographics are not available.

Usually domains with no demographics are these where the identifier is seen as the pseudonym. But this is not always the case. There are cases where an identifier is linked to a person and at the same time demographics of that person are not available. An example for such a case is the domain of health professionals. In that case, the eHealth ID of a health professional is not a pseudonym, but access to the demographics of the health professional is not necessary available at the source.

This case is important, as a pseudonymisation of identifiers from a domain without demographics is generally possible (e.g., to pseudonymise the eHealth ID of health professionals). Since such identifiers are registered at the National Pseudonymisation Service without any demographics, a link to an existing person is never possible.

To stay in the example: Pseudonyms of health professionals are never linkable to pseudonyms of patients, even if the health professional and the patient are the same person.

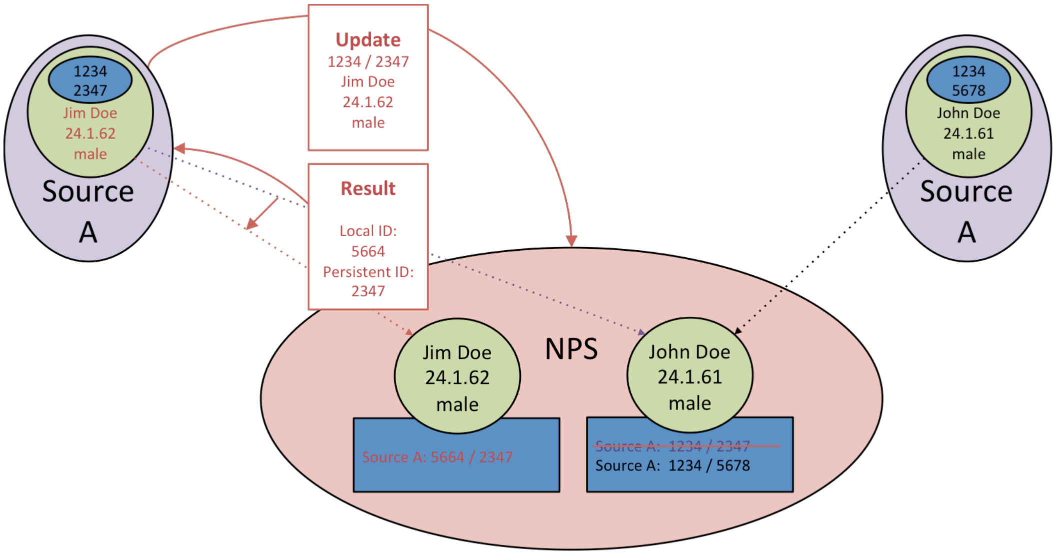


Figure 6. Problematic case: Update affects source (after update)

I. *Identity Linking*

If a source figures out that two local identifiers belong to the same person (even if the demographics are different), the source can perform a linkage-request to tell the National Pseudonymisation Service that one local identifier will never be used anymore and that all data that is linked to the obsolete identifier will belong to the surviving local identifier. Properties and permissions

Figure 7 gives an overview about the relationship of properties and rights concerning systems and identifier: A system, e.g., server, client application or user, belongs to one or more identifier domains. For a specific domain it is defined whether a system has certain permissions or not:

Provide demographics:

Not all systems inside a domain should be allowed to provide demographics at the National Pseudonymisation Service. Some systems are only allowed to use the local identifier inside that domain.

Update demographics:

In case of first contact, some systems must be allowed to provide demographics to the National Pseudonymisation Service. Update of demographics is a critical task that only should be permitted to some selected systems.

Link identifier:

Similar to update of demographics, the linking of identifier is a rare case that only should be done after the identification of doublets in the local system is beyond question.

Retrieve demographics:

This is the most critical task in the whole concept of the National Pseudonymisation Service. Retrieval of demographics on base of a given local identifier should only be possible in rare cases, e.g., re-identification of persons in case of important notifications.

For reasons of data protection, the National Pseudonymisation Service will only provide the latest version of demographics that has been provided by a system in that domain. Demographic details from other domains will not be accessible. Also this permission will only provide data, if the domain itself manages demographics. Since domains that only have access to pseudonyms never provide demographics to the National Pseudonymisation Service, the retrieval of demographics in that domain is excluded.

For a specific domain, properties define whether it is a source domain with demographics or a destination domain with pseudonyms:

Demographics available:

In domains with demographics available, a source domain is given. Usually, in domains without demographics, this is not the case (except in a relation 1st level pseudonym, 2nd level pseudonym).

Identifier managed by source:

For source domains, it has to be defined, whether a local identifier is managed inside the domain, or if it has to be provided by the National Pseudonymisation Service. In the second case, it must be defined, whether a persistent identifier is used to manage update conflicts or if always a new local identifier will be used in that case.

For destination domains without demographics, the National Pseudonymisation Service will always manage the local identifier. It is not possible that the domain itself manages pseudonyms.

Number range of local identifier:

In *Section G.* Identifier domain and identifier number space it is explained, why the National Pseudonymisation Service only manages numbers as local identifiers. This property defines the range of the number space.

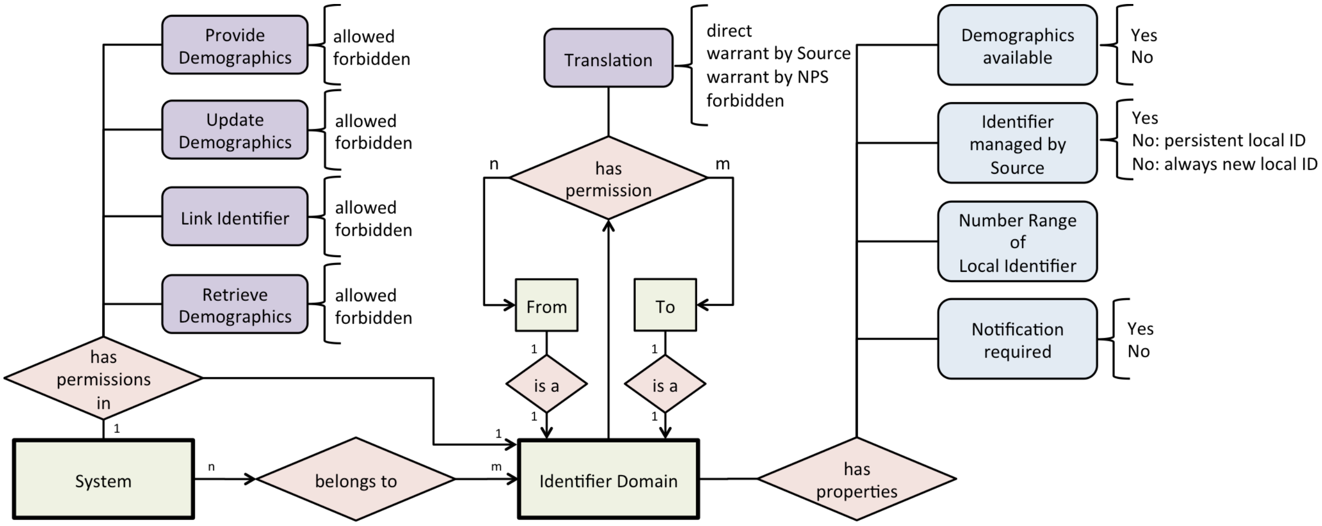


Figure 7. Properties and permissions

Notification required:

Systems might fail or crash in the wrong moment. Some tasks might require notification to ensure that the involved systems have stored the result of a request in their databases. If a system notifies a certain result, the responsibility for the use of the information is moved from the National Pseudonymisation Service to the notifying system.

In the use cases that are described in *Section B.* Main cases of data transmission, all cases have a from-to relation in regards to the translation of local identifiers or the creation of warrants. These relations need to be defined as permissions for direct or warrant-based translations in the National Pseudonymisation Service.

Case B:

A source system from a source domain has permission to translate its local identifier (From-Domain) directly to the destination domain (To-Domain).

Case C:

A system from a destination domain has permission to translate the local identifier from a source domain (From-Domain) directly to its local identifier (To-Domain).

Case D:

A source system from a source domain has permission to translate its local identifier (From-Domain) directly to a third domain (To-Domain), and a system from a destination domain has permission to translate the local identifier from a third domain (From-Domain) directly to its local identifier (To-Domain).

Case E:

A direct translation of local identifiers is not permitted, so a translation requires the use of a warrant. A system of the source domain (From-Domain) has permission to create a warrant (Warrant by Source) or retrieve a warrant (Warrant by NPS) for the destination domain (To-Domain).

The data model of Figure 7 allows the definition of permissions that are not useful: In the direct translations the system must either belong to the from-domain or to the to-domain. In the warrant-based translation, the permission, the system must belong to the from-domain.

J. Identity related services

As a result from the previous sections, the following services are required at the National Pseudonymisation Services. Services that are needed in the in-house setup are explicitly named. For simplification reasons, persistent local identifiers are mentioned in most of the description, but it depends on the definition of the domain, whether a persistent identifier has to be used or if it will be returned or not.

Services for the notification of the reception of identifiers and warrants are not listed.

All functions require the "Identifier Domain" parameter. This parameter is needed to identify the current domain of the calling system, since systems might belong to several domains.

1) Register a person by Demographics

```
Register Person
     Identifier Domain
     Demographics
→    Local/Persistent Identifier
```

Returns a local identifier on base of demographics.

2) Register a person by demographics and local identifier

```
Register Identified Person
     Identifier Domain
     Local Identifier
     Demographics
```

In domains that manage the local identifiers by themselves, the service makes the National Pseudonymisation Service aware of the local identifier and its associated demographics in that identifier domain. No persistent identifiers are provided in self-managed domains. This function does not return any result.

3) Update of demographics of a person

```
Update Person
     Identifier Domain
     Local/Persistent Identifier
     Demographics
→    Local Identifier
```

If a local identifier has already been registered at the National Pseudonymisation Service, or the National Pseudonymisation Service has returned a local/persistent identifier, this function is used to update the demographics. This might lead to an update of the local identifier (see *E.2)*.

4) Translate identifer at the source domain

```
Translate Identifier
     Local Identifier Domain
     Foreign Identifier Domain
     Local/Persistent Identifier
→    Foreign/Persistent Identifier
```

This is a simple translation of identifiers between the local (source) and the foreign (destination) domain.

This is mainly the function that is needed in the in-house setup, so the local identifier of the foreign domain is calculated or determined on base of the local identifier only.

5) Trananslate identifier at the destination domain

```
Retrieve Identifier
     Local Identifier Domain
     Foreign Identifier Domain
     Foreign/Persistent Identifier
→    Local/Persistent Identifier
```

This service is similar to 4) but in this case, the destination domain is calling the service. This leads to a change of the focus of the local-foreign relation: the destination is requesting its local identifier on base of the foreign identifier of the source.

6) Register a warrant, associated to a local identifier

```
Register Warrant
     Local Identifier Domain
     Foreign Identifier Domain
     Local/Persistent Identifier
     Warrant
```

This function registers a warrant for a foreign domain with a given local identifier. In this case the warrant is managed (provided) by the source

The warrant is only valid in the foreign domain to retrieve the local identifier of that domain.

7) Request a warrant, associated to a local identifier

```
Request Warrant
     Local Identifier Domain
     Foreign Identifier Domain
     Local/Persistent Identifier
→    Warrant
```

This function requests a warrant for a foreign domain with a given local identifier. In this, case the warrant is managed (provided) by the National Pseudonymisation Service.

The warrant is only valid in the foreign domain to retrieve the local identifier of that domain.

8) Retrieval of the local identifier at the foreign domain on base of a warrant

```
Redeem Warrant
     Local Identifier Domain
     Warrant
→    Local/Persistent Identifier
```

Having a warrant of the correct domain, this service will allow the retrieval of the identifier in that domain.

9) Re-identification of demographics on base of a local identifier

```
Re-Identify Person
     Local Identifier Domain
     Local/Persistent Identifier
→    Demographics
```

In case of re-identification requests, this service will only provide the latest version of demographics that have been registered in that domain. Demographics from different domains related to the same persons are not accessible.

This service might also be useful in the in-house setup in case of re-identification requests.

10) Linking of local identifiers, in case of identified doublets

```
Link Local Identifier
     Local Identifier Domain
     Obsolete Local Identifier
     Surviving Local Identifier
```

If a source that manages the local identifiers itself identifies doublets, should use this function to inform the National Pseudonymisation Service about the merge in the local (in-house) system.

11) Get updates of identifiers in the domain

```
Get Updates
     Local Identifier Domain
→    List of
     [Persistent Identifier: New Local Identifier]
```

In case of updates at the National Pseudonymisation Service, local identifiers might change for some data (see *E.* Problematic cases). The National Pseudonymisation Service is a passive service so it only responses to requests. Identifier domains must use this service regularly to get notified about the latest updates, since the last request.

12) Identification of potential duplicates

```
Vigilance Request
     Local Identifier Domain
     Local/Persistent Identifier 1
     Local/Persistent Identifier 2
```

On base of the medical data, a destination domain might come to the conclusion that the given local identifiers are potential duplicates and belong to the same person. An alert will be triggered at the identity vigilance of the National Pseudonymisation Service to check the case.

13) Identification of potential splits

```
Vigilance Request
     Local Identifier Domain
     Local Identifier
     Persistent Identifier 1
     Persistent Identifier 2
```

On base of the medical data, a destination domain might come to the conclusion that the given local and persistent identifiers are potential splits and should belong to different persons. An alert will be triggered at the identity vigilance of the National Pseudonymisation Service to check the case.

K. Matching of identities

The National Pseudonymisation Service uses an underlying Master Patient Index to figure out, whether the given demographics of a person are known (match), or if they identify an unknown person (no-match). The matching algorithm depends on mandatory demographics (first name, last name, gender, and birthday) and optional demographics (national social security number, zip-code of the birthplace).

Depending on the degree of agreement, the algorithm will distinguish, true matches (the person is known with high probability), true non-matches (the person is not known with high probability), and ambiguous matches (there is more than one potential candidate or it is not clear whether the person is known or not).

If the decision is not clear (ambiguous match), a new person will be created in the system, and the identity vigilance will be informed to solve the problem by requesting additional information from the involved domains.

Since the National Pseudonymisation Service acts as a shell around an existing Master Patient Index, the Master Patient Index service could be replaceable at any time in case without affecting the pseudonymisation service.

L. Calculation of local identifiers

An important part of the entire process of identification of persons is the creation of the local identifiers of a domain. This calculation has to be done at the National Pseudonymisation Service or locally at the in-house solution for new persons or for persons that are accessed for the first time by an identifier domain. Domains that provide their own local identifier are not affected by this question.

Each person that is managed by the National Pseudonymisation Service (or internally by its Master Patient Index) is represented by a person-object. This object consist of the single best record of the demographics of the persons plus an internal identifier of the object. Each local identifier of an identifier domain is linked to that object via the internal object identifier. The link will be established during the registration step of the person or the translation of identifiers between different domains. If an identifier does not exist at that time, it must be created.

In the in-house solution, usually the managed persons are stored inside a database with a person identifier associated to it. This might be an attribute of the database table or it is a given identifier that was inscribed together with the demographics (e.g., social security number) or it was already a pseudonym that was given with the data. If personal data needs to be delivered to a certain domain, the domain specific identifier needs to be created, if this has not been done already.

In both cases there are several options to create the identifier out of the person identifier (object identifier, person identifier, pseudonym, social security number etc.):

Take the next free available number: last used number plus 1:

In this case all created numbers build a continuous running number. This must be avoided, if the identifier is used as a pseudonym. If the original identifiers are already continuous numbers, a link could be established between identifier and time of creation of the person inside the system.

Chose a random number:

The use of random numbers should be the preferred choice, but require the management of mapping tables (local identifier → person identifier).

Such mapping tables could be used in case of selective anonymisation of individuals: If the entry (local identifier → person identifier) is replaced with (local identifier → NULL) Every data that is stored with the local identifier can never be linked to the person again.

Calculate the identifier from the person identifier:

If the management of mapping tables must be avoided (especially in the in-house setup) and a selective anonymisation is not required, the calculation of the local identifier on base of the person identifier together with a certain secret is a good alternative to the random number.

Good strategies are the use of salted hashes (`Hash(Salt + person identifier)`) or encryption (`Enc(Key, person identifier)`). In both cases, the salt or the key is the secret that is linked to the identifier domain.

This strategy is problematic, if the calculated local identifier has limitations related to the data type. Example: The person identifier at the source is of data type *4 byte unsigned integer* (=32 bit), and the resulting local identifier must be from the same data type.

Current hashing or encryption algorithms usually work with 128 bit minimum, so are not suitable in the described case. Cropping of the result to 32 bit is not a way to go because this introduces a risk of collisions, which means that for some person identifier the calculated local identifier will be the same. This behavior cannot be tolerated. For this special case, a new calculation algorithm is proposed.

M. Calculation of local identifier with small number of bits

The mathematics behind the local identifier calculation of a person identifier is based on selected primitive roots of fixed prime numbers as it is used in the Diffie-Hellman protocol to ensure a secure key exchange [15]. First we need to introduce some fundamental mathematics.

1) Discrete logarithm

Having the equation:

$$b = a^i \mathit{mod}\, p, \text{ with } p \text{ prime}, i \in \{1..p\text{-}1\} \qquad (1)$$

Then i is called the discrete logarithm. This is equivalent to

$$i = \log_a b \mathit{mod}\, p, i \in \{1..p\text{-}1\} \qquad (2)$$

The calculation of b is easy but currently there exists no efficient way to find the discrete logarithm i with given a, b and p.

This statement is only true if p is big enough to make the use of pre-calculated solution tables impossible and if no pre-knowledge about i exists that allows reducing the search space.

2) Primitive roots

The property of a being a primitive root of prime p means that

$$a^i \; mod\, p, \; with\; i = 1..p\text{-}1 \qquad (3)$$

results in all values of 1..p-1, with no value double or missing. This property is relevant to create collision free local identifiers.

Primitive roots have been used already a long time ago to build good random number generators [19]. The proposed algorithm uses this knowledge to introduce pseudo-randomness into the series of pseudonyms.

3) Adaption for the calculation of the local identifier

With k bits that are reserved for the local identifier, a prime number p should be chosen that in best case is the highest prime number lower than 2^k. With the given p, the interval of possible person and local identifiers is 1..p-1. The numbers that are invalid in the k-bit number space are 0 and $p..2^k$-1. As an example: For k=31, the highest prime lower than 2^{31} is 2^{31}-1. In this case, only 0 and 2^{31}-1 cannot be used as person and local identifier.

The difficulty to find the discrete logarithm i of the equation a^i *mod p* is based on the assumption that i is randomly distributed and that no information can be used to reduce the number of possible values. This may not be the case if the persons person identifier is used as exponent i.

Two examples might help to demonstrate the problem. In both cases, i equals the person identifier *id*.

In the first example the exponent i is a continuous number starting with 1, so the n^{th} local identifier belongs to the person identifier n. If an attacker is able to estimate the number of already managed persons, the number of potential i is heavily reduced.

In the second case, the person identifier is created out of the birthday and a running number (e.g., 1985032312 for the 12th person born in March 23 of 1985). In the example, knowing that a person was born at a certain day, this limits the number of potential i to 100.

To avoid the reduction of potential i with prior knowledge about the person identifier *id*, two processing-steps are performed, including one non-linear step:

1. XOR (non-linear exclusive or):
   The person identifier will be XORed with a constant $c \neq 0$ of k bits
2. EXPAND:
   The intermediate result is multiplied with an expansion factor q mod p, $(1<q<p)$

Step 1 might lead to an invalid results that is out of the range of the allowed values (0, $p..2^k$ -1). If this happens the XOR must be reversed. In case of p be close to 2^k, the number of invalid values (p..2^k-1) can be minimized, which lowers the risk to revers the XOR step.

p being prime guarantees that the result of step 2 is still in the range of 1..p-1, avoiding any doubles.

At that point, even with pre-knowledge about the person identifier, no conclusions about the exponent i of the calculation a^i *mod p* can be made, which would allow to reduce the search space. Finally, the main calculation step a^i *mod p* can be performed.

Unfortunately, if the prime number p is small, it is possible to calculate all possible $b=a^i$ *mod p* to set up a solution table $b \mapsto i$. For a prime smaller than 2^{31}, maximal 8GiB are needed to setup such a table (1GiB = 2^{30} Byte). Even for prime smaller than 2^{40}, a solution table with maximal 5TiB needs to be pre-calculated (1TiB = 2^{40} Byte). Tables with that size fit in currently used RAM or hard disks and are no burden for potential attackers. A solution to overcome this problem is to also keep the primitive root a secret. In that case, with given b and p, for each a a different i exists that fulfills the equation.

The entropy of the secrets a, q and c that have been used so far might be insufficient to avoid brute force attacks. So a final round of confusion is performed:

3. XOR (non-linear exclusive or):
   The intermediate result will be XORed with a constant $d \neq 0$ of k bits
4. ROL (shift rotate left):
   The intermediate result will be shift-rotated s bits left (|s|>0)

As with step 1, step 3 must be reversed, if the result is invalid. If the intermediate result of step 4 leads to an invalid value, it must be repeated until the intermediate result is in the allowed range. Both strategies do never introduce duplicates.

The calculated local identifier finally is the outcome of step 4. Figure 8 lists the entire algorithm as pseudo code.

The complexity of an attacker to re-identify the person ID is based on the secrets a, c, d, q and s and requires knowledge about some person and local identifier pairs to proof if the secrets are correctly identified.

4) Example

All calculation steps of the local identifier for the person identifier *id*= 300568 are shown in Figure 9.

- Let k=31 and prime $p=2^{31}$-1=2147483647.
- The initial value of *id* will be XORed with c=1656294509.
- The expansion factor is defined as q=41795.
- a=572574047 is a primitive root from p.
- The intermediate result will be XORed with d=913413943.
- Finally, an intermediate result will be shift-rotated left with s=11 bits.
- The pseudonym that has been calculated from this identifier is 353489627.

5) Finding a primitive root

For a given prime number p it is unnecessary to find all primitive roots to select the secret a; only one primitive root

```
FUNCTION calculateLocalIdentifer (id, k, a, p, c, d, q, s)
BEGIN
   t1 := id XOR c                // XOR person identifier
                                 // with secret c
   IF (t1 ∉ {1 .. p-1}) THEN     // if out of range
      t2 := id                   // reverse if necessary
   END IF
   t2 := (t1 * q) mod p          // expand with secret p
   i := t2                       // this is the exponent

   b := a^i mod p                // the main calculation

   t3 := b XOR d                 // XOR with secret d
   IF (t3 ∉ {1 .. p-1}) THEN     // if out of range
      t3 := b                    // reverse if necessary
   END IF
   t4 := t3 ROL s                // shift-rotate-left s bits
   WHILE (t4 ∉ {1 .. p-1}) DO    // if out of range
      t4 := t4 ROL s             // repeat if necessary
   END WHILE
   lid := t4                     // the local identifier
   RETURN lid
END
```

Figure 8. Pseudocode of the algorithm

is needed. The density of primitive roots is quite high so it requires approximately four random tries in case of p=2^{31}-1 until a primitive root is found. To proof if a selected a is a primitive root, the series of a^i mod p (i=1..p-1) has to be checked. If a^i mod p = 1 $with$ i≠p-1, the series can be stopped and a is not a primitive root. In that case two exponents are found resulting in the same value: a^{i+1} mod p = a = a^1 mod p.

The series can easily be calculated with

$$a^0 \mathit{mod}\ p = 1 \qquad (4)$$

$$a^i \mathit{mod}\ p = a(a^{i-1} \mathit{mod}\ p)\ \mathit{mod}\ p\ \mathit{for}\ i=1..p-1 \qquad (5)$$

This is a quite time consuming process. A faster way to go is this:

First all prime factors of p-1 have to be identified. In case of p=2^{31}-1, the prime factors of 2^{31}-2 = 2147483646 are 2, 3, 7, 11, 31, 151, and 331. The time to identify the prime factors has only to be spent once and does not affect the time to test the primitive root candidates.

For each prime factor f from p-1 the values a^i mod p with i=(p-1)/f need to be calculated. a is a primitive root of p if none of the results equals 1. In the example the series of $a^{2147483646/2}$ mod p, $a^{2147483646/3}$ mod p, $a^{2147483646/7}$ mod p, ..., $a^{2147483646/331}$ mod p needs to be calculated. These are maximal seven calculations.

6) Calculating a^i mod p

For the calculation of a^i mod p in the described algorithm, the pre-calculation of a^{i-1} mod p is not available; so, the recursion as mentioned in the equations (4) and (5) is not applicable. Alternatively, the calculation can be quickened if i is split into its binary representation of k bits:

$$i = (i_{k-1}, i_{k-2}, \dots, i_2, i_1, i_0)\ with\ i_j \in \{0,1\} \qquad (6)$$

```
t1   = id XOR c
     = 300568 XOR 1656294509 =
     = 1656593013

t2   = (t1 · q) mod p
     = (1656593013 · 41795) mod 2147483647
     = 284715408

b    = a^t2 mod p
     = 572574047^284715408 mod 2147483647
     = 465777933
t3   = b XOR d
     = 465777933 XOR 913413943
     = 766681658
t4   = t3 ROL s
     = 766681658 ROL 11
     = 353489627

lid = t4
     = 353489627
```

Figure 9. Example calculation

$$i = \sum_{j=0}^{k-1} 2^j \cdot i_j \ \ with\ i_j \in \{0,1\} \qquad (7)$$

Then

$$a^i\ mod\ p = \qquad (8)$$

$$a^{\sum_{j=0}^{k-1} 2^j \cdot i_j}\ mod\ p = \qquad (9)$$

$$\left(\prod_{j=0}^{k-1} a^{2^j \cdot i_j} \right) mod\ p \qquad (10)$$

This calculation is very fast in case of pre-calculated $a^{2^j}\ mod\ p$ using

$$a^{2^0}\ mod\ p = a \qquad (11)$$

$$a^{2^j}\ mod\ p\ =\ (a^{2^{j-1}}\ mod\ p)^2\ mod\ p \quad for\ j=1..k-1. \qquad (12)$$

As an example, let i = 25 = 11001_2. Then

$$a^{25}\ mod\ p\ = \qquad (13)$$

$$\left(a^{2^4 \cdot 1} \cdot\ a^{2^3 \cdot 1} \cdot a^{2^2 \cdot 0} \cdot a^{2^1 \cdot 0} \cdot a^{2^0 \cdot 1}\right) mod\ p = \qquad (14)$$

$$\left(a^{2^4} \cdot\ a^{2^3} \cdot a^0 \cdot a^0 \cdot a^{2^0}\right) mod\ p = \qquad (15)$$

$$\left(a^{2^4} \cdot\ a^{2^3} \cdot 1 \cdot 1 \cdot a^{2^0}\right) mod\ p = \qquad (16)$$

$$\left(a^{2^4} mod\ p \cdot\ a^{2^3} mod\ p \cdot a^{2^0} mod\ p\right) mod\ p \qquad (17)$$

7) Bit-depth of the secrets

The algorithm for the calculation of the local identifiers is useless, if the used secrets allow a brute-force attack. This is not the case, if the entropy of the used secretes is big enough. Furthermore, the effort to calculate the pseudonym must allow the calculation of a high number of pseudonyms per time.

Several secrets to calculate the pseudonym are used:

TABLE I. FACTS

	4-byte signed integer	5-char base64 6-char base32	2-byte signed short integer
Bits	32	30	16
maximal positive value	2^{31}-1	2^{30}-1	2^{15}-1
highest possible prime	2^{31}-1	2^{30}-35	2^{15}-19
highest possible person identifer	2 147 483 646	1 073 741 789	32 748
number of invalid values	2	36	20
number of possible primitive roots of the prime	534 600 000	459 950 400	10 912

The number of possible primitive roots can be calculated with Eulers φ-function and is *φ(φ(p))= φ(p-1)*.

- The random number *c* that was used to XOR the exponent.
- The factor *q* that was used to expand the exponent.
- The primitive root *a*.
- The random number *d* that was used to XOR the intermediate result.
- The number of ROLs (left-shift-rotate) of the intermediate result *s*.

As an example, the bit-depth of the secrets are calculated in case of data types that are usually used to store person identifiers

- 4-Byte signed integer:
  The number space is sufficient for a third of the entire living population on earth or four times the number of the living population of the European Union.
- 2-byte signed short integer:
  The number space is only useful for a small set of persons, e.g., for persons of a clinical study.
- 5 chars of base64-encoded numbers or 6 chars of base32-encoded numbers
  (in case of efficient human readability):
  The number space is sufficient for two times of the living population of the European Union but insufficient for the living population the People's Republic of China.

With the information of Table I, the entropy of the secrets can be calculated that are used during the calculation (Table II).

For integer and the encoded char-values, the secret with entropy of ≈124 bits is sufficient to avoid effective brute force attacks. This is void for short integer. Here the entropy of the secrets is only ≈64 bits. In that case, the calculation of the pseudonym must be performed in two rounds with different primitive roots, expansion factors, XORs and shift values. This does not fully double the entropy of the secrets because the final steps XOR and ROL are directly followed by another XOR step of the next round. All three steps can be simplified to only one XOR plus ROL. However, the entropy of the secret (≈111 bits) is sufficient today.

TABLE II. ENTROPY OF THE SECRET

Secret	4-byte signed integer	5-char base64 6-char base32	2-byte signed short integer
a: primitive roots	≈ 29 bit	≈ 29 bit	≈ 13 bit
q: expansion factor	≈ 31 bit	≈ 30 bit	≈ 16 bit
c: XOR exponent	31 bit	30 bit	15 bit
d: XOR result	31 bit	30 bit	15 bit
s: ROL result	≈ 5 bit	≈ 5 bit	≈ 4 bit
total	*≈127 bit*	*≈ 124 bit*	*≈ 63 bit*

8) *Calculation speed*

There are only a few steps involved in the calculation of the pseudonym. The calculation of a^i *mod p* is identified as the most time consuming calculation. The calculation is straightforward and avoids several rounds until the final result is available. Multiplications are always more time consuming than XOR or shift operations so it is assumed that the pseudonym calculation is slower that the competitive approaches. In the known scenarios, the number of pseudonymisation calculations per time is sufficient: Tests have shown that on average hardware (Intel Core 2 Duo, 2.66 GHz) 132.5-thousand pseudonyms per second can be calculated.

9) *Attacks*

Important for the evaluation of the algorithm is the resistance against attacks and the possibility for re-identification.

It is known that for *b* = a^i *mod* p (*p* prime, *a* primitive root of *p*) it is difficult to calculate the discrete logarithm *i*, if *b*, *a,* and *p* are known and *p* being big enough to avoid solution tables. In our case, also the primitive root *a* is unknown. On the other hand, there might be pre-knowledge about *i*. With the non-linear diffusion steps that base on the use of non-trivial secrets (e.g., $q{\neq}1$, $c{\neq}0$), the exponent is complex enough to make the information of the initial series useless.

Brute force attacks will only be possible if an attacker is able to validate the set of parameters with a given set of person identifiers and their associated local identifiers. An attacker will in worst case only get both sets, not knowing what person identifier and local identifier is finally linked. Depending on the size of the set, it is likely that several secret sets lead to the same transformation of the set of person IDs to the set of pseudonyms. In case of leaked pairs of person plus local identifier, this information can only be used to perform a brute force attack. A recalculation of the used parameters is not possible.

10) *Re-Identification*

A fast re-calculation of the person identifier is possible if all secrets are known. In case of small *p* and a given *a*, the solution table for *b*=a^i *mod p* is made fast and every step of the entire calculation process can be reversed.

Only if the solution table cannot be pre-calculated, it is quicker to pseudonymise all known person identifiers again to find the correct local identifier.

III. Results

A National Pseudonymisation Service on base of the described concept has been specified and is in the final phase of implementation in Luxembourg. The concept was developed after an intensive study of the demands has been carried out in Luxembourg. The National Pseudonymisation Service creates a shell around the National Master Patient Index that will be used in the National eHealth Platform of Luxembourg. This ensures, that the National Pseudonymisation Service will cover all persons working or living in Luxembourg and that all persons are managed with high quality demographics. Matching difficulties of identities should therefore be an exception.

Identity vigilance in case of uncertainty will be covered on a national level and no double structures have to be created. The use of the National Master Patient Index by the National Pseudonymisation Service does not affect the productivity of the used system. Both systems can be enhanced independently and update paths do not affect each other.

The described functions and the possibility to adapt the properties of an identifier domain for several needs, allows the use of the National Pseudonymisation Service in all Cases from B to E as described in Section *B*.

A. Case B: Cancer Register using National Pseudonymisation Service

The use of the National Pseudonymisation Service as described in Case B is planned for a cancer register.

In the described use case, the sources have access to the clinical data of the patients and will send pseudonymized extracts of this data to the cancer register. Sources can be divided into sources that manage their local identifier, and those who do not manage a local identifier.

The process of sending data from the sources to the cancer register can be described as follows:

1a) Sources with managed local identifer register local identifer and demographcis at the National Pseudonymisation Service

```
Register Identified Person
     Source Domain (Managed)
     Local Identifier
     Demographics of Patient
```

1b) Sources with unmanaged local identifer request local identifer on base of demographics from the National Pseudonymisation Service

```
Register Person
     Source Domain (Unmanaged)
     Demographics of Patient
→    Local/Persistent Identifier
```

2) Request the pseudonym of the cancer register from the National Pseudonymisation Service

```
Translate Identifier
     Source Domain (Managed/Unmanaged)
     Cancer Register Domain
     Local/Persistent Identifier

→    Pseudonym/Persistent Identifier
         of Cancer Register
```

3) Source sends medical data and pseudonym to the cancer register

```
Send Medical Data
     Pseudonym/Persistent Identifier
         of Cancer Register
     Medical Data
```

4) National Cancer Register stores medical data and pseudonym

```
Store Medical Data
     Pseudonym/Persistent Identifier
         of Cancer Register
     Medical Data
```

B. Case E: Biobank using National Pseudonymisation Service

A Luxembourgish biobank currently uses the principles of Case E with the use of a Trusted Third Party as pseudonymisation service. The migration of that service to the National Pseudonymisation Service is planned as soon as the service is available. The specialty of this concept is the uses of the warrant. In the biobank case, cryo-boxes are sent by the biobank to the collection sites. If samples are collected from donors (specimen, blood, urine) the samples are put into the cryo-box that is sent back to the biobank. The kit-ID of the cry-box acts as the warrant in the process of person identification and pseudonym retrieval.

The process can be described as follows:

1) The biobank sends cryo-boxes with unique kit-IDs to the collection sites

```
Send Cryo-Box
     Kit-ID
```

2) Collection sites with unmanaged local identifer request local identifer on base of demographics from the National Pseudonymisation Service

```
Register Person
     Collection Site Domain
     Demographics of Donor

→    Local Identifier
```

3) Collection Sites take samples of donors and stores it into cryo-boxes

```
Collect Samples
     Cryo-box with Kit ID
     Samples of a Donor
```

4) Collection Sites send cryo-boxes to biobank

```
Send Cryo-Box
     Cryo-box with Kit ID
     Samples of a Donor
```

5) Collection Site registers Kit-ID of the cryo-box as warrant at the National Pseudonymisation Service

```
Register Warrant
     Collection Site Domain
     Biobank Domain
     Local Identifier
     Kit-ID
```

6) Biobank request pseudonym at the National Pseudonymisation Service on base of the Kit-ID of the received cryo-box

```
Redeem Warrant
         Biobank Domain
         Kit-ID
→        Pseudonym/Persistent Identifier
```

7) Biobank stores samples in its repository and links it in its Laboratory Information Management System (LIMS) with the pseudonym

```
Store and Manage
         Sample of Donor
         Pseudonym/Persistent Identifier
```

C. Case B: HIV Register using In-House Pseudonymisation

A local HIV register performs long-term studies on HIV. It was created several years ago and recently introduced the concept of in-house pseudonymisation to improve data privacy and data security. Since all tools and mechanisms had been implemented around an existing database structure, it was decided to keep the original database with all the medical data plus the demographics of the patient untouched. Persons who have direct contact to the patients fill this database.

A tool is used to create pseudonymized copies of the original database that contain only research and study specific subsets of the original data. Therefore, the database model is only a subset of the original database model, but it is ensured that none of the used tools have to be adapted. Such extraction, transform and load tools are called ETL tools. The ETL tool will handle all mappings between both database models and finally will create the pseudonym out of a given person identifier.

In the described case, the keeping of mapping tables (person identifier-to-random pseudonym) was not wanted, and the described techniques of hashing or encryption have also not be suitable, since the data type that the person identifier and pseudonym is 4 byte signed integer with 1 as the smallest, and 2^{31}-1 as the highest possible values.

With the algorithm that has been described in *II.M.* Calculation of local identifier with small number of bits, study specific pseudonyms are calculated out of the given person identifier. For each study, a different set of secrets (as listed in Table II) is used as the calculation parameters.

Since the identification of primitive roots is not an easy task, a tool was provided to identify primitive roots.

D. Use cases in discussion

Other Luxembourgish institutes are highly interested in using a National Pseudonymisation Service in the near future, either to secure their existing databases or for newly planned databases.

For some of the analyzed use cases, the use of a National Pseudonymisation Service seems to be far too much and an in-house pseudonymisation is demanded.

IV. Conclusion and Future Work

The use of a National Pseudonymisation Service solves several problems of researchers. It divides infrastructure and personal costs among all users of the national service. It ensures the quality of the underlying demographics that is ensured by the existence of a centralized identity vigilance that already exists for the underlying National Master Patient Index of the National Pseudonymisation Service. The team that performs identity vigilance on national level has permission to solve unclear matching decisions of the Master Patient Index by questioning all sources of demographics. Update of demographics and reconsiderations at the National Pseudonymisation Service about the matching of persons are manageable. With the use of the persistent identifiers, only selective data needs to be updated in case of change of identifiers.

In case of new studies or trials that have to be approved by ethics commissions, questions about data protection will be asked. The use of mechanisms that already have been accepted on a national level will simplify the answering of these questions.

If approved by the ethics commission and with given consent by patients, a National Pseudonymisation Service enables the exchange of data between different studies or trials and link data from the different sources to the same person, even if the sources only use pseudonyms.

The given set of services and the various properties that can be configured for an identifier domain allows the implementation of all described cases A to E. There are always arguments pro and contra the implementation of a certain case, depending on risks to disclose sensitive information. Each designer of a clinical study or trial setup can decide, which of the cases suits most his requirements and data privacy demands.

Even with an up and running National Pseudonymisation Service, the use of in-house pseudonymisation might be the first choice, especially in case of limited participants in the setup. In that case the use of a national service might be far too much, and the costs for the service might be too high. In that case the described algorithm for the creation of pseudonyms out of person identifiers provides a collision free one-way pseudonymisation technique for small bit-depth that still fulfills the requirements of a one-way function, if the secrets behind the calculation are kept secret.

The past has shown that an up-and-running National Pseudonymisation Service improves the willingness to include pseudonymisation solutions already during the design phase of new research databases. This is good, since privacy-by-design strategies are more durable than security patches that are introduced in a later phase of development.

It is expected that with the establishing of the National Pseudonymisation Service, local companies will link their software solution to the national service. Alternatively, consultant companies will offer help in the planning of the integration of the National Pseudonymisation Service into future applications and to find the correct setup (case A to E) that suites most the demands of the customer on data protection and disclosure risks.

Acknowledgment

This material is based upon work supported by the Agence eSanté of Luxembourg.

References

[1] U. Roth, "Protecting the privacy with human-readable pseudonyms: One-way pseudonym calculation on base of primitive roots," *Proceedings of the Sixt International Conference on eHealth, Telemedicine, and Social Medicine, eTelemed 2014*, pp. 111–115, 2014.

[2] B. Alhaqbani and C. Fidge, "Privacy-preserving electronic health record linkage using pseudonym identifiers," *10th International Conference on e-health Networking, Applications and Services, HealthCom 2008*, pp. 108–117, 2008.

[3] B. Riedl, V. Grascher, S. Fenz, and T. Neubauer, "Pseudonymization for improving the privacy in e-health applications," *Proceedings of the 41st Annual Hawaii International Conference on System Sciences, HICSS 2008*, p. 255, 2008.

[4] ZorgTTP, "Transparency builds trust," 2012. [Online]. Available from: https://www.zorgttp.nl/userfiles/Downloads/ZorgTTP-englishbrochure-2012.pdf 2014.11.30

[5] K. Pommerening and M. Reng, "Secondary use of the EHR via pseudonymisation," In: *L. Bos, S. Laxminarayan, A. Marsh (eds.): Medical Care Compunetics 1*, pp. 441–446, IOS Press, 2004.

[6] B. Morris, P. Rogaway, and T. Stegers, "How to encipher messages on a small domain," *29th Annual International Cryptology Conference, CRYPTO 2009, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, pp. 286–302, LNCS 5677, Springer 2009.

[7] E. Stefanov and E. Shi, "FastPRP: Fast pseudo-random permutations for small domains," *IACR Cryptology ePrint Archive, Report 2012/254*, 2012. [Online]. Available from: http://eprint.iacr.org/2012/254 2014.11.30

[8] S. Dara and S. Fluhrer, "FNR: Arbitrary length small domain block cipher proposal," *Security, Privacy, and Applied Cryptography Engineering, 4th International Conference, SPACE 2014, Pune, India, October 18-22, 2014. Proceedings*, pp. 146–154, LNCS 8804, Springer 2014.

[9] J. Daemen and V. Rijmen, "The design of Rijndael," Springer-Verlag New York, Inc., 2002.

[10] D. Eastlake 3rd and T. Hansen, "US secure hash algorithms (SHA and SHA-based HMAC and HKDF)," Request for Comments 6234, RFC 6234 (Informational), 2011.

[11] T. Ristenpart and S. Yilek, "The mix-and-cut shuffle: Small-domain encryption secure against N queries," *33th Annual International Cryptology Conference, CRYPTO 2009, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pp. 392–409, LNCS 8042, Springer 2013.

[12] A. Faldum and K. Pommerening, "An optimal code for patient identifiers," *Computer Methods and Programs in Biomedicine,* vol. 79, no. 1, pp. 81–88, 2005.

[13] T. Xie, F. Liu, and D. Feng, "Fast collision attack on MD5," *IACR Cryptology ePrint Archive, Report 2013/170*, 2013. [Online]. Available from: http://eprint.iacr.org/2013/170 2014.11.30

[14] L. R. Rivest, A. Shamir, and L. Adleman,"A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM,* Vol 21 (2), pp. 120–126, ACM, 1978.

[15] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory,* vol. 22, no. 6, pp. 644–654, IEEE Press, November 1976.

[16] IHE ITI Technical Committee, "IHE IT infrastructure technical framework supplement: Patient identifier cross-reference HL7 V3 (PIXV3) and patient demographic query HL7 V3," IHE, August 10, 2010.

[17] IHE Wiki. *Patient identifier cross-referencing*. [Online]. Available from: http://wiki.ihe.net/index.php?title=Patient_Identifier_Cross-Referencing 2014.11.30

[18] ITU T-HDB-LNG.4-2010. *Object identifiers (OIDs) and their registration authorities*. [Online]. Available from: http://www.itu.int/pub/T-HDB-LNG.4-2010 2014.11.30

[19] S. K. Park and K. W. Miller, "Random number generators: good ones are hard to find," *Commun. ACM,* vol. 31, no. 10, pp. 1192–1201, 1988.

An Evaluation Framework for Adaptive Security for the IoT in eHealth

Wolfgang Leister
Norsk Regnesentral
Oslo, Norway
wolfgang.leister@nr.no

Mohamed Hamdi
School of Communication Engineering
Tunisia
mmh@supcom.rnu.tn

Habtamu Abie
Norsk Regnesentral
Oslo, Norway
habtamu.abie@nr.no

Stefan Poslad
Queen Mary University
London, UK
stefan.poslad@qmul.ac.uk

Arild Torjusen
Norsk Regnesentral
Oslo, Norway
arild.torjusen@nr.no

***Abstract*—We present an assessment framework to evaluate adaptive security algorithms specifically for the Internet of Things (IoT) in eHealth applications. The successful deployment of the IoT depends on ensuring security and privacy, which need to adapt to the processing capabilities and resource use of the IoT. We develop a framework for the assessment and validation of context-aware adaptive security solutions for the IoT in eHealth that can quantify the characteristics and requirements of a situation. We present the properties to be fulfilled by a scenario to assess and quantify characteristics for the adaptive security solutions for eHealth. We then develop scenarios for patients with chronic diseases using biomedical sensors. These scenarios are used to create storylines for a chronic patient living at home or being treated in the hospital. We show numeric examples for how to apply our framework. We also present guidelines how to integrate our framework to evaluating adaptive security solutions.**

Keywords*—*Internet of Things; evaluation framework; scenarios; assessment; eHealth systems; adaptive security.

I. Introduction

Wireless Body Sensor Networks (WBSNs) improve the efficiency of eHealth applications by monitoring vital signs of a patient using low-rate communication media and constitute an important part of the Internet of Things (IoT) by bringing humans into the IoT. However, the successful deployment of the IoT depends on ensuring security and privacy, which need to adapt to the processing capabilities and resource use of the IoT. To evaluate such adaptive mechanisms we introduced evaluation scenarios specifically designed for applications in eHealth and proposed an evaluation framework [1]. This evaluation framework is extended in this study with a quantitative component that allows us to quantify the quality of security solutions.

The "Adaptive Security for Smart Internet of Things in eHealth" (ASSET) project researches and develops risk-based adaptive security methods and mechanisms for IoT that will estimate and predict risk and future benefits using game theory and context awareness [2]. The security methods and mechanisms will adapt their security decisions based upon those estimates and predictions.

The main application area of ASSET is health and welfare. Health organisations may deploy IoT-based services to enhance traditional medical services and reduce delay for treatment of critical patients. In a case study, we evaluate the technologies we developed for adaptive security using both simulation and implementation in a testbed based upon realistic cases. Blood pressure, electrocardiogram (ECG) and heart rate values can be gathered from patients and anonymised. The sensor data can be stored in different biomedical sensor nodes that are capable of communicating with any of the following connectivity options ZigBee, Wi-Fi, 3G, GPRS, Bluetooth, and 6LoWPAN. For instance, a smartphone with a suitable transceiver could act as an access point between sensor nodes and a medical centre. For the evaluation, we developed a set of scenarios to assess the adaptive security models, techniques, and prototypes that will be introduced in ASSET. These scenarios describe the foreseeable interactions between the various actors and the patient monitoring system based on IoT.

In computing, a scenario is a narrative: it most commonly describes foreseeable interactions of user roles and the technical system, which usually includes computer hardware and software. A scenario has a goal, a time-frame, and scope. Alexander and Maiden [3] describe several types of scenarios, such as stories, situations (alternative worlds), simulations, story boards, sequences, and structures. Scenarios have interaction points and decision points where the technology under consideration can interact with the scenario. This means that the scenarios developed for a particular situation have to take into consideration the technologies used by the different actors. The importance of scenarios in the assessment of security solutions has been discussed in the literature [4], [5]. This work focuses on the development of scenarios that support the evaluation of adaptive security techniques for the IoT in eHealth.

There are many definitions of the IoT. For instance, while the ITU-T [6] defines the IoT as "a global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies", the European Research Cluster on the Internet of Things (IERC) defines the IoT as "A dynamic global network infrastructure with self-configuring capabilities based on standard and interoperable communication protocols where physical and virtual *things* have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network" [7]. For our purposes we use Abie and Balasingham's shorter definition: "IoT is a network of things" [2]. Habib and Leister [8] present a review of IoT layer models, including the ITU-T IoT reference model [6].

The primary contributions and advances of this study are the development of a quantitative framework for the assessment of adaptive security solutions on the basis of security, privacy, Quality of Service (QoS) requirements, and costs.

In Section II, the requirements and the proposed assessment framework are described including metrics that make this framework quantifiable in order to enable comparison of various situations. We define the properties that must be fulfilled by a scenario to assess adaptive security schemes for eHealth. We show the interaction between the scenarios, the threats, and the countermeasures in an assessment framework for the ASSET project.

In Section III, we describe the extension of a previously developed generic system model, which is used for the structure of the scenarios in Section III-A with different QoS requirements, contexts and adaptive security methods and mechanisms. These scenarios, first proposed by Leister et al. [9], include a patient monitored at home scenario, a hospital scenario, and an emergency scenario. These scenarios are reviewed and their adequacy to the evaluation of adaptive security techniques for the IoT is analysed. We propose storylines that can support requirements analysis, as well as adaptive security design, implementation, evaluation, and testing.

Further, in Section IV, we present storylines for both the home monitoring scenario and the hospital scenario. These storylines are used in Section V to show how our framework can be applied to selected episodes of the home scenario and storyline. In Section VI, we show how to use our framework in the context of adaptive security as defined by Abie and Balasingham [2]. Finally, Section VII discusses our framework and relates it to other work before Section VIII offers concluding remarks and future prospects.

II. The ASSET Evaluation Framework

Designing the scenarios is of central significance for the ASSET project. They depict the operation of systems, here applied to IoT-based eHealth systems, in the form of actions and event sequences. In addition, scenarios facilitate the detection of threats and the identification of the solutions to cope with these threats. In a scenario-based assessment, a set of scenarios is developed to convey the design requirements. With regard to the specific objectives of IoT-based systems, the scenarios should capture two types of requirements:

1) *Security requirements:* Novel adaptive security and privacy mechanisms and methods are required to adapt to the dynamic context of the IoT and changing threats to them. Thus, the scenarios should be generic enough to capture the security needs for the data processed and exchanged within a patient monitoring system. This is particularly challenging because this system encompasses multiple networking technologies, data, users, and applications, addressing varying processing capabilities and resource use.

   In an assessment context, privacy and security requirements are related. Privacy addresses the ability to control the information one reveals about oneself over the Internet and who can access that information.

2) *QoS requirements:* QoS addresses the overall performance of a system regarding technical parameters. Unlike many traditional applications and services relying on communication networks, eHealth applications have stringent QoS requirements. Items such as the communication delay, the quality of the communication channels, and the lifetime of the self-powered sensor nodes are crucial context parameters that have significant impact on the safety of the patient. The scenarios should highlight the needs in terms of QoS requirements and illustrate the dynamic interplay between these needs and the security requirements.

Security and QoS mechanisms are interrelated. Adaptation of security mechanisms may impact the QoS and vice-versa. QoS requires adaptive security mechanisms to ensure appropriate level of QoS. While adapting poor security mechanisms can hamper the performance of QoS, an inappropriate QoS level can leak sensitive information about the importance of the service in question. Therefore, adaptation must consider both security and QoS together to achieve the best possible security and QoS levels. Otherwise, weaker security and/or less effective QoS guarantees may be the result. For example, the requirement of using stronger cryptographic algorithms could have negative impact on the performance or battery consumption.

A. Requirements and Sets of States

The ASSET scenarios appear as a component of an assessment framework that will serve to improve the applicability of the security techniques proposed in the frame of the project. The other components of the assessment framework are (*i*) a set of threats describing the actions that violate the security requirements, (*ii*) a set of security solutions that mitigate the threats, and (*iii*) a set of system states representing the dynamic context in which the patient monitoring system operates. Fig. 1 illustrates the ASSET assessment framework. The security and QoS requirements are the output of the scenario design activity. In other terms, the scenarios should give information about the set of reliable states from the security requirements, here denoted as $\mathcal{S}$, and the set of states where the QoS is acceptable, here denoted as $\mathcal{Q}$. The intersection of these sets is the set of desirable states, denoted in Fig. 1(a) by $\mathcal{D}$ (Desirable), where the security and QoS requirements are balanced.

One of the intrinsic features of the ASSET scenarios is that the sets of security requirements and QoS requirements could vary in time and space. This will make the threats and the security solutions also vary in time and space. Threats

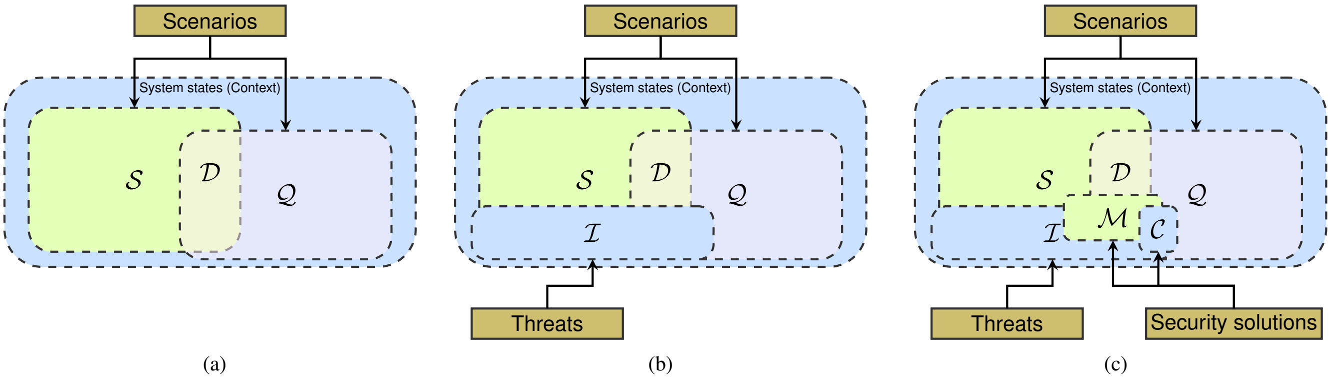


Fig. 1: The ASSET assessment framework.

are viewed as potential events that may generate insecure system states, while countermeasures are intended to thwart the effects of these threats. The realisation of a threat reduces the set of secure states in the scenario of interest and affects the QoS. This is represented by the region $\mathcal{I}$ (Impact) in Fig. 1(b). This region represents a set of states that will not fulfil the security or QoS requirements if a threat is realised. The countermeasures or *controls* [10] will reduce both the likelihood of a threat being realised and the impact of an emerging threat. Hence, the size of the set of potentially insecure states is decreased. Fig. 1(c) illustrates the effect of the countermeasures through the Region $\mathcal{M}$ (Mitigate). This region extends the set of secure states. Nonetheless, the countermeasures can have a negative effect on the QoS, represented by the region $\mathcal{C}$ (Cost), consisting of power, processing resources, memory, communication overhead, and cases where QoS requirements may not be fulfilled.

These elements are used in a scenario-based assessment framework to evaluate the strength of the adaptive security solutions. For instance, the scenarios allow us to evaluate the strength of the security controls to minimise the impact of threats in a given context.

For adaptive security solutions, the proposed protection techniques will vary in time and space according to the context. This is not conveyed by the scenario representation of Fig. 1. To overcome this issue, we derive a set of storylines from the ASSET scenarios. These can be viewed as a sequential application of the scenarios in a way that the selection of the appropriate countermeasures must take into consideration:

- *The space transition between scenarios.* Space encompasses much useful information that affect the security decision-making process. For instance, the location of the WBSN may increase/decrease its vulnerability. Moreover, mobility introduces significant challenges including horizontal and vertical handover management, i.e, managing handover on the same layer or within the same access technology and between different layers or different access technologies, respectively.
- *The time transitions between scenarios (with its implications on the context).* The time interplay between the threats and countermeasures has a substantial and dynamic impact on the environment where the patient monitoring system is deployed. The amount of energy, memory, and processing resources are crucial parameters from the QoS perspective and the security solutions have to adapt accordingly. In addition, the state of the communication channel and the proper temporal interplay in all these contexts are important in the selection of the appropriate security decisions.

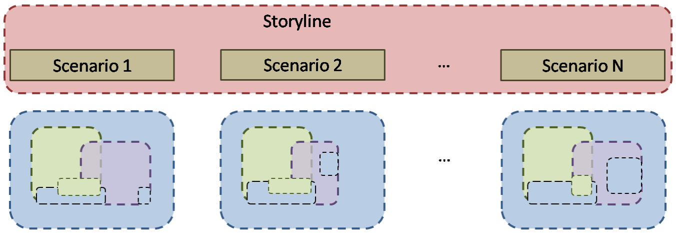


Fig. 2: Illustration of context changes during the execution of a storyline. The use of the different shaded regions follows that of Fig. 1

Fig. 2 illustrates the evolution of the storyline and the underlying impact on the context. Of course, the sequence of scenarios forming a storyline should be consistent so that it translates to a real-case situation.

B. Making the ASSET Framework Quantifiable

Assessing the qualities of a given system state can be done by means of data given by human assessors and by means of objective data from measurements. Our goal is to establish an estimation function that takes measured data as input and which is a prerequisite to implement functionality for adaptive security. To establish such an estimation function the assessment a panel of users and specialists is queried to calibrate a function that uses measured data as input. Similar methodology has been used to estimate the quality of streamed video [11]. In the following we present how to assess a given system state by using human assessors.

To make the ASSET framework quantifiable we define a real function $0 \leq q(\text{system state}) \leq 1$ that shall express the degree of how well the requirements are fulfilled in the system state in question. A low value, below a given threshold, denotes that the system state in question is unacceptable, while a value close to 1 denotes that most requirements are well fulfilled.

The function q is composed of three parts: *1*) security requirements that need to be fulfilled, expressed in the function q_S; *2*) degree of fulfilled QoS requirements, expressed in the function q_Q; and *3*) costs that occur due to mitigation of threats. The function q is then composed of a product of all partial functions of $q_{i \in \{S,Q,C\}}$: $q = \prod_i {q_i}^{w_i}$. The weights are real numbers $0 \leq w_i < \infty$ and express the importance of a single q_i, large values indicating more importance. A weight $w_i = 1$ is considered neutral. The importance of each parameter is defined by the assessor according to the nature of the requirement before assessing the q_i values.

The above definition has the disadvantage that the resulting q is sensitive to the number k of factors q_i that are used to define it. To mitigate this we propose to replace the weights by $v_i = \frac{w_i}{\sum_{j=1}^{k} w_j}$ resulting in $\hat{q}_i = {q_i}^{\frac{w_i}{\sum_{j=1}^{k} w_j}}$. Thus, the value q is expressed by:

$$q = \prod_i \hat{q}_i = \prod_i {q_i}^{\frac{w_i}{\sum_{j=1}^{k} w_j}} \qquad (1)$$

1) Security Requirements: Define $\mathcal{G_S} = (\mathcal{S} \setminus \mathcal{I}) \cup \mathcal{M}$ as a set of states where security requirements are fulfilled or threats are mitigated. For states j outside $\mathcal{G_S}$ we define a deviation from the ideal requirements and a normalised distance d_{S_j} : $0 \leq d_{S_j} \leq 1$ according to a suitable metric to denote how far the current state is from ideal fulfilment of the requirement. We set $d_{S_j} = 1$ when deviations cannot be tolerated. Thus, we define the following function:

$$q_{S_j} = \begin{cases} 1 & \text{if state } \in \mathcal{G_S} \\ 1 - d_{S_j} & \text{if state } \notin \mathcal{G_S} \end{cases}$$

2) QoS Requirements: Define $\mathcal{G_Q} = \mathcal{Q} \setminus \mathcal{C}$ as a set of states where all QoS requirements are fulfilled and possible effects from the mitigation are tolerable. For states j outside $\mathcal{G_Q}$ we define a deviation from the ideal QoS requirement and a normalised distance d_{Q_j} : $0 \leq d_{Q_j} \leq 1$ according to a suitable metric to denote how far the current state is from ideal fulfilment of the requirement. We set $d_{Q_j} = 1$ when QoS requirements are insufficiently fulfilled.

QoS requirements may be unfulfilled due to influences from the environment, or become unfulfilled due to adaptation. The latter could, for instance, happen if a security requirement to avoid eavesdropping was met by reducing signal strength, which could impact the available bandwidth or even data availability.

Thus, we define the following function:

$$q_{Q_j} = \begin{cases} 1 & \text{if state } \in \mathcal{G_Q} \\ 1 - d_{Q_j} & \text{if state } \notin \mathcal{G_Q} \end{cases}$$

3) Mitigation Costs: Besides the effect on QoS there may be other costs implied by mitigation, e.g., real costs in payroll or material, changes to the environment, costs for the patient, virtual costs for a lower QoS, and so on. States with unacceptable costs are included in the area $\mathcal{C}$. For costs outside $\mathcal{C}$ we define relative costs on a normalised scale $d_C : 0 \leq d_C \leq 1$. We define the following function:

$$q_C = \begin{cases} 1 - d_C & \text{if costs } \notin \mathcal{C} \\ 0 & \text{if costs } \in \mathcal{C} \end{cases}$$

C. Assessment to define the q_i values

To aid human assessors in assessing the values for q_i (i.e. the value indicating how far a given requirement is from the ideal fulfilment) we propose to base the assessment on a set of questions that are evaluated based on a Likert scale [12]. A Likert scale is a psychometric scale commonly involved in research that employs questionnaires where the questions are to be answered from *best* to *worst* on a scale of n steps, where n is an odd integer number.

If the questionnaire to be filled out by an assessor is designed so that each q_i corresponds to one question on a Likert scale we propose to use a function e that takes the response $\tilde{q}_i \in \mathbb{N}$ for $0 \leq \tilde{q}_i \leq n - 1$ as an argument. We use two approaches to express the q_i.

1) Linear Approach:

$$q_i = e_\alpha(\tilde{q}_i) = \frac{\tilde{q}_i}{n-1} \qquad (2)$$

2) Logarithmic Approach:

$$q_i = e_\beta(\tilde{q}_i) = \log_n(\tilde{q}_i + 1) \qquad (3)$$

Using the logarithmic approach leaves less impact of bad values than the linear approach. There are some caveats on using a logarithmic function for values on a Likert scale, as noted by Nevill and Lane [13]. Particularly, the values on the Likert scale should express a continuous and rather equidistant increase of quality.

3) Other Methods: In case the questionnaire is designed in a way that several independent questions result in one value for q_i, Bayesian networks developed by Perl and Russell [14] can be employed. However, we consider the design of the questionnaires and the use of Bayesian networks as future work. Note also that for Bayesian networks more data from an assessment are necessary than for the above mentioned methods.

While the Likert scale is useful for assessing opinions on a psychometric scale, i.e., subjective data, we need, as well, to be able to assess objective data. In these cases, we set up a scale where discrete choices on a questionnaire are mapped to a similar scale as the Likert scale to reflect the quantity of data based on an objective value. This way of creating assessment data are quite common for assessments, such as in the estimation of the quality of software products in the OpenBRR [15, 16].

When objective data are used as input, e.g., as the result of measurements, these data on a continuous scale can be mapped

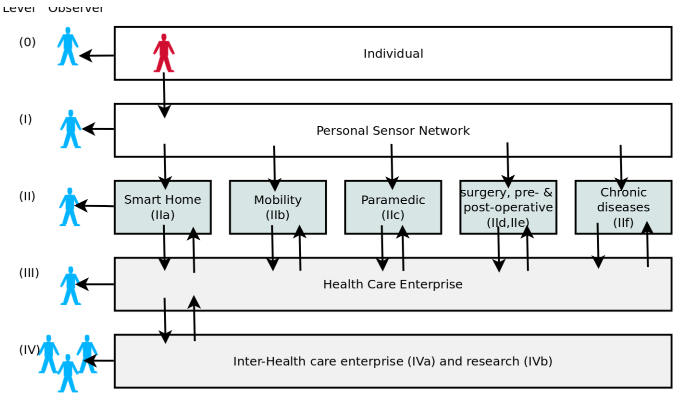


Fig. 3: Generic eHealth framework indicating the use cases in five levels (Extended from [17]).

into the value range $0 \leq q_i \leq 1$ and used in eq. (1). Note, however, that the mapping function does not necessarily need to be linear, and a specific assessment phase may be necessary to develop a suitable function that maps the values into the value range $0 \leq q_i \leq 1$.

4) Assessment by Subject Panels: For an assessment often several individuals are put into an assessment panel. These subjects perform the assessment individually while the results are put together into one assessment result. Further work needs to show whether it is more practicable to calculate individual q values and then calculate some mean value of these or whether to calculate mean values for each $\tilde{q}_i$.

III. Extended Generic Model for eHealth Scenarios

In the following sections, we develop the scenarios of the ASSET project and show how storylines can be extracted. We also underline the role of the storyline in the assessment of adaptive security techniques for eHealth. Before delving into the details of scenario and storyline engineering, we highlight the major properties that a scenario should have in order to be useful for evaluating adaptive security.

Patient monitoring systems are a major data source in healthcare environments. During the last decade, the development of pervasive computing architectures based on the IoT has consistently improved the efficiency of such monitoring systems thereby introducing new use cases and requirements. It is important that these monitoring systems maintain a certain level of availability, QoS, and that they are secure and protect the privacy of the patient. Previously, we have analysed the security and privacy for patient monitoring systems with an emphasis on wireless sensor networks [17] and suggested a framework for providing privacy, security, adaptation, and QoS in patient monitoring systems [18]. We divided patient monitoring systems into four Generic Levels (GLs): (0) the patient; (I) the personal sensor network; (II) devices in the closer environment following several scenarios; and (III) the healthcare information system.

We review the generic model presented by Leister et al. [18] and extended by Savola et al. [19]. This extended generic model contains three new levels related to the monitoring of chronic diseases, the communication between multiple healthcare providers, and the communication between healthcare providers and medical research institutions, respectively. Consequently, the extended generic model is composed of five levels numbered from (0) to (IV) depending on the logical distance to the patient to whom Level (0) is assigned. Multiple types are considered at Level (II). Note that only one of these types applies at a time. However, it must be possible to switch between the types in Level (II) depending on the activity of the patient. To this purpose, the communication between Levels (II) and (III) is two-way. The key levels of our extended generic model are as follows, as shown in Fig. 3:

(0) **Patient.** This is the actual patient.

(I) **Personal sensor network.** The personal sensor network denotes the patient and the sensors measuring the medical data. These sensors are connected to each other in a WBSN. While this sensor network can be connected randomly, in most cases one special WBSN node is appointed to be a Personal Cluster Head (PCH), which forwards the collected data outside the range of the WBSN.

(IIa) **Smart home.** The patient is in a smart-home environment where the personal sensor network interacts with various networks and applications within this environment. The smart home infrastructure may be connected to a healthcare enterprise infrastructure using long-distance data communication.

(IIb) **Mobility.** The patient is mobile, e.g., using public or personal transportation facilities. The personal sensor network of the patient is connected to the infrastructure of a healthcare enterprise via a mobile device, e.g., a mobile Internet connection.

(IIc) **Paramedic.** The WBSN is connected to the medical devices of an ambulance (car, plane, and helicopter) via the PCH. The devices of the ambulance can work autonomously, showing the patient status locally. Alternatively, the devices of the ambulance can communicate with an external healthcare infrastructure, e.g., at a hospital.

(IId) **Intensive care/surgery.** During an operation the sensor data are transferred to the PCH or directly to the hospital infrastructure over a local area network. The sensors are in a very controlled environment, but some sensors may be very resource limited due to their size, so extra transport nodes close to the sensors may be needed.

(IIe) **Pre- and postoperative.** During pre- and postoperative phases of a treatment, and for use in hospital bedrooms, the sensor data are transferred from the sensor network to the PCH and then to the healthcare information system.

(IIf) **Chronic disease treatment.** The WBSN data are used by healthcare personnel in non-emergency treatment of individual patients with a chronic disease.

(III) **Healthcare information system.** This is considered a trusted environment. It consists of the hospital network, the computing facilities, databases, and access terminals in the hospital.

(IVa) **Inter-healthcare provider.** Information is shared between different healthcare providers concerning medical information of an individual patient.

(IVb) **Healthcare provider and research.** Information is shared between healthcare providers and medical research organisations for the purposes of research, new solutions development, etc.

A. The Structure of the Scenarios

Through the potential interactions between these levels, notice that the model can support the elaboration of multiple scenarios where the actors interact by switching from a level to another. The scenarios in healthcare using biomedical sensor networks are quite complex. Therefore, they need to be efficiently structured. We consider three main scenarios (hereafter denoted as *overall scenarios*) and we decompose them into sub-scenarios (hereafter denoted as *core scenarios*). A particular interest is given to the transitions between the core scenarios since these transitions constitute substantial sources of threats. Here, we consider three scenarios, a home scenario A shown in Fig. 4, a hospital scenario B shown in Fig. 6, as well as an emergency scenario C.

Each of these overall scenarios contain a set of core scenarios which are denoted by the scenario identifier A, B, or C, followed by a dash and the core scenario numbering using roman numbers. The transitions between these core scenarios model the interaction between the various components of the patient monitoring system. In this paper, we focus mostly on Scenario A where the patient is supposed to be monitored outside the hospital while performing normal daily actions. To extract useful technical cases for the evaluation phase we need to structure the scenario according to the patient's actions and situation.

TABLE I shows a list of core scenarios used in our work, which overall scenario they belong to, and which transitions are useful. Note that other transitions are theoretically possible, but these are either unlikely or can be achieved by combining a series of transitions, e.g., taking Core Scenario A-ii (moving) as an intermediate for Overall Scenario A. Omitting unlikely transitions helps to reduce the number of states when modelling the scenarios.

TABLE I: Overview of core scenarios. The bullets mark scenarios that are part of the respective core scenario.

		scenario			transition to
core scenario & name		A	B	C	core scenario
i	home monitoring	●			ii, xiv
ii	moving	●			i, iii, iv, viii, vi, v
iii	public transport	●			ii
iv	vehicle transport	●			ii
v	shop	●			ii
vi	café	●			ii
vii	doctor's office	●			ii, xiv
viii	waiting room	●	●		vii, ix, ii
ix	diagnosis		●	●	x, xi, xii, ii
x	operation		●		xi
xi	intensive care		●		xii
xii	observation		●		ii, xi, ix
xiii	accident			●	xiv
xiv	ambulance			●	ix

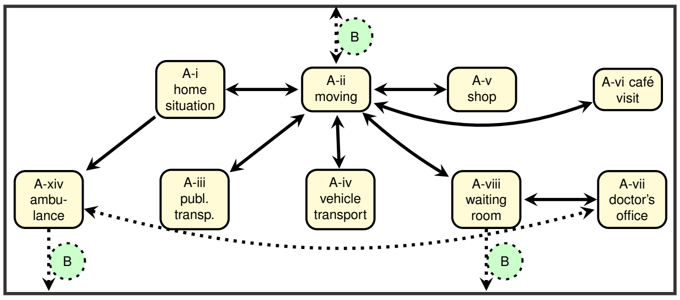


Fig. 4: The Home Scenario with the underlying core scenarios and their transitions.

B. The Structure of the Home Scenario

In Scenario A, a monitored patient can be in various contexts performing normal daily actions. For example, for a patient with diabetes the following situations apply:

- The patient is at home or a nursing home using monitoring equipment.
- The patient uses sensors and communicates electronically with the doctor's office.
- The patient uses specific monitoring equipment for diabetes.
- The patient visits the doctor's office regularly and uses public transport or a car to get there.
- At the waiting room the patient can communicate data to the health care infrastructure of the doctor's office.
- The patient regularly takes walking or jogging trips.
- The patient regularly visits a café with friends; this includes walking or commuting with public transport.
- In case of an emergency or planned surgery, not necessarily related to her condition, the patient may be sent to a hospital with an ambulance.

This list of situations is not yet a useful narrative. It needs to be structured and enriched with the specific context information, such as the necessary devices of the IoT, the communication channels, and actions of the involved actors. This is done in the core scenarios that describe a specific part of an overall scenario; e.g., a situation a patient experiences. Each core scenarios can be part of several overall scenarios.

1) Home Situation (monitored at home) (A-i): Biomedical sensors are employed in an environment where the patient is at home or in a nursing home. The patient is monitored by a WBSN, and the sensor data and alarms can be transmitted to medical centres and emergency dispatch units. The patient uses a smartphone with health-diary software that also imple-

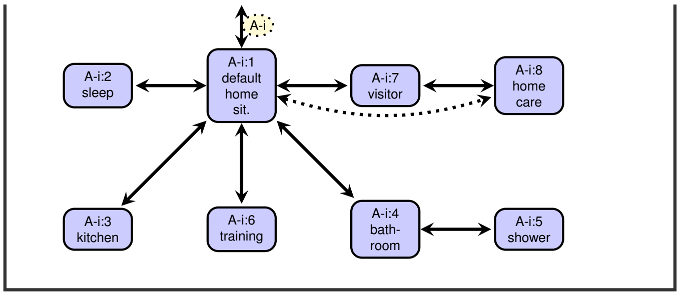


Fig. 5: The detail-scenarios of the home-situation.

ments personal health records (PHR) and stores measurements continuously.

Here, the sensors may not be monitoring or transmitting the physiological patient data continuously in order to reduce battery power consumption. Instead, depending on a predefined algorithm, abnormal sensor data from certain sensors may activate an alarm that is sent to a central monitoring unit.

On a regular basis, the patient transmits measurements to the medical information system at the doctor's office, thus synchronising the PHR with the medical information system; the patient also has an audio-/video-conversation where medical questions are discussed. During these sessions the patient may take pictures with the smart phone camera or perform other measurements.

In this scenario, the following characteristics are given:

1) Ease of use and non-intrusiveness are important issues.
2) Very low power consumption, enabling a long life span of the batteries, is required.
3) A network infrastructure is available, such as access to the Internet via LAN, WLAN, or mobile networks.
4) Limited mobility, handoff is possible, but infrequent.
5) Privacy and observability of signals are important requirements.

Core Scenario A-i can be split up into several detail-scenarios that may depend on the patient's activities, time of the day, or context, as shown in Fig. 5. These sub-scenarios may include the generic scenario (A-i:1), sleeping (A-i:2), kitchen work (A-i:3), visiting the bathroom (A-i:4), taking a shower (A-i:5), training (A-i:6), receiving a visitor (A-i:7), or receiving a home care nurse (A-i:8). All these detail scenarios create different challenges regarding security and QoS that need to be addressed by adaptive security methods. For example, when taking a shower, the sensors may need to be unmounted, while receiving visitors may create the need to give access to selected data or devices.

2) Moving (Walking, Jogging, Cycling) Scenario (A-ii): The patient does daily training, i.e., jogs in the nearby park, or does shorter walks from the home to the public transport, to the café, shop, or doctor's office. A common feature in these situations is that the patient needs to use a smartphone as a device that collects sensor data, using the mobile networks to transmit the data. When walking, jogging, or taking a bicycle ride in the park many other people and their devices may interfere with the communication of the smartphone.

When walking in the woods, there may be several spots which are not covered by a mobile network. In this case, the signal is so weak that only emergency calls from another provider will work. While data traffic is not possible, SMS messages can be used to send data with very low bandwidth, possibly after several retries. For an average walking trip, this outage may last for some minutes. However, SMS is asynchronous and messages may take minutes to days to arrive. Thus, it may be quicker to wait until the user, if still mobile, moves to a region where there is network coverage.

3) Transport Scenarios: We consider two transport scenarios, one with public transport, and one with commuting by car.

Core Scenario A-iii presents a situation where a patient commutes to a doctor's office or to a café using public transport. Here, the patient needs to use a smartphone as a device that collects sensor data, using the mobile networks to transmit the data. Blind spots without connectivity to a mobile network, roaming, varying data transmission quality, etc., are parts of this scenario. This scenario can be applied to long-distance trains, planes, etc.

In Core Scenario A-iv the patient uses her own or another's (private) car to commute to a shop, a café, or the doctor's office. Here, the patient needs to use a smartphone as a device that collects sensor data, using the mobile networks or networks installed or used in the car to transmit the data. Blind spots without connectivity to a mobile network, roaming, varying data transmission quality, etc., are parts of this scenario.

4) Shop Scenario (A-v): Another situation defined by Core Scenario A-v is when the patient is in a shop. In addition to the conditions of A-vi, the patient is given the opportunity to check groceries to be compliant with the patient's diet and allergy-prohibition plans, access information from the shop, and use a shopping list.

5) Café Scenario (A-vi): The patient visits a café. Here, the patient needs to use a smartphone as a device that collects sensor data, using mobile networks or café's WLAN zone for data transfer. Switching between the WLAN and mobile networks may occur, the WLAN may be of varying quality, many other café visitors may interfere, or the WLAN may not actually be connected to the Internet.

6) Doctor's Office Scenario (A-vii): The patient is in the doctor's office, usually after some time in a waiting room (A-viii). Here, the patient can have extra sensors attached. These extra sensors, as well as the existing sensors, can communicate with the doctor's infrastructure either through the smartphone of the patient, or directly, depending on the needs. A doctor can change a sensor's characteristics, which requires the possibility to re-program the sensor devices.

7) Waiting Room Scenario (A-viii): The patient is in a waiting room at a doctor's office or a hospital. Patients that are known to the healthcare system can be connected from their smartphone to the healthcare network; here, specific actions for collecting data from the device or other preparations can be performed. Once the patient is in the range of the waiting room, the smartphone can transfer large amounts of stored patient data directly to the infrastructure of the medical centre via short-range communication, instead of using long-range mobile communication.

8) Other scenarios: In the scenario structure we foresee that the patient can undergo a transition to other core scenarios in a different overall scenario in order to cover situations that else would be outside the scenario structure. For instance, a patient could get ill and be brought to a hospital in an ambulance (B-xiv) or an emergency situation happens (Scenario C). Note that the use of devices in the IoT could be different in Scenarios A, B, and C: as an example, in an emergency situation the use of one of the patient's own sensors would not be possible in all cases.

C. The Structure of the Hospital Scenario

In Scenario B, the biomedical sensors are used in a hospital environment. Here, the patient is located in an operating room (OR) or intensive care unit (ICU) while undergoing intensive monitoring of vital physiological parameters. Additional sensors may be required during this procedure to monitor other physiological parameters. The patient may be moved between different rooms during the treatment, e.g., from the OR to the ICU, but monitoring must continue. The sensor data may need to be transferred over different wireless networks. The system should be able to cope with a breakdown in sensor nodes, new software updates, wireless network traffic congestion, and interferences from other wireless networks and biomedical devices.

In Scenario B, a fixed network infrastructure is available between Levels (II) and (III) which can be accessed by the sink nodes of the biomedical sensor network. The scenario includes a complex communication environment. Interference from co-existing wireless networks, mobile networks, and various medical facilities is possible; this may reduce the performance of the transmission. The network topology in this scenario is fixed, but changes to the network topology may happen while patients are moving or being moved from one place to another, possibly causing handoff to other gateways. However, roaming to other networks is not part of this scenario in order to stay within the hospital domain.

Note that scenarios that seem to be similar in Scenario B and in Scenario A may have differences that are not obvious. Thus, one cannot use reasoning performed in one scenario in another without checking the context and other conditions. For instance, A-vii (doctor's office) could be different from a similar situation in a hospital (B-ix) since the hospital is connected to a different kind of network infrastructure. Usually, the primary healthcare points (doctor's office) and hospitals have different security requirements and policies.

1) Hospital Diagnosis Scenario (B-ix): The patient is examined; extra sensors are attached, and existing sensors on the patient may be accessed both directly and via the patient's smartphone. In addition, NFC tags are used to identify objects. The medical personnel can re-configure and re-program the sensors during diagnosis.

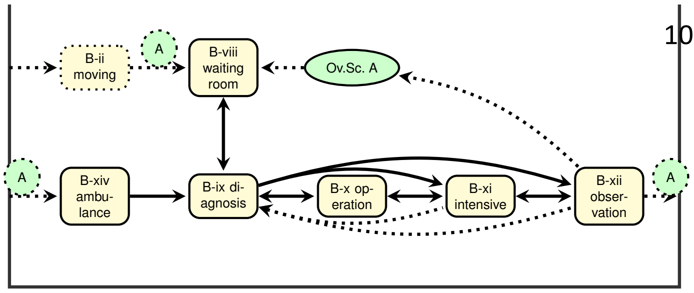


Fig. 6: The Hospital Scenario with the underlying core scenarios and their transitions.

2) Hospital Operation Scenario (B-x): The patient is undergoing surgery; extra sensors are attached, and existing sensors on the patient are accessed directly by the hospital system rather than through the smartphone of the patient. In this scenario, the QoS is set very high, while security-wise the sensors are in a protected zone. The medical personnel can re-program the sensors during the operation.

3) Hospital Intensive Care Scenario (B-xi): The patient is in intensive care after an operation. Extra sensors are attached, and existing sensors on the patient may be accessed both through the patient's smartphone, and directly through the hospital infrastructure. In addition, NFC tags are used to identify objects. In most cases, the smartphone will be used as PCH. The medical personnel can re-program the sensors during intensive care.

4) Hospital Observation Scenario (B-xii): The patient is in a room under "normal" observation; in contrast to the home situation, the patient's smartphone has direct access to the hospital systems and will deliver data directly with higher QoS through the secured hospital systems.

D. The Structure of the Emergency Scenario

The Emergency Scenario (C) presents an emergency situation where victims are provided with sensors, patients are transported with an ambulance (car, helicopter, plane) and delivered to the emergency reception at a hospital. In Scenario C the use of sensors is not planned beforehand, health personnel must improvise, the identity of the patient may be unknown, and the infrastructure may be partially unavailable. Despite this, the expectation is that severely injured patients are stabilised, and they survive the transport to the emergency reception in the best condition possible.

We include the first scenario of the Hospital Scenario, the diagnosis phase when the patient arrives in Core Scenario C-ix. Here, the rather unplanned interventions at the emergency site are adapted to the routines at the hospital.

1) Accident Site Scenario (C-xiii): This scenario is a disaster and accident response scenario where biomedical sensors are deployed to measure values such as blood pressure, temperature, pulse and ECG in an ad-hoc network at the site of an accident. Wired or wireless communications infrastructures may be damaged or unavailable, and a large number of severely injured people may overwhelm the emergency field personnel. This could prevent them from providing efficient and effective emergency rescue. Biomedical sensor networks can be quickly deployed to monitor vital signs. A large number of injured can be monitored simultaneously.

In this scenario, the following characteristics are given:

1) The sensor network must operate autonomously, and needs a high degree of self-organisation. The network topology is highly dynamic. Therefore, the sensor nodes should be able to discover each other and setup a sensor network autonomously.
2) A fixed network infrastructure is not available; data transferred from Level (II) to Level (III) must use a mobile network or other specific wireless network, such as microwave, or digital trunk communication.
3) The radio link may be unstable and the radio link quality may vary. Additionally, the communication environment is rather complex, since many sensor nodes may be deployed in a small area, possibly causing severe channel competition.
4) High degree of mobility. Handoffs are possible and may be frequent.
5) Blue-light functionality. That is, being able to re-use sensors on short notice with high flexibility (short-cutting some of the usual procedures).

2) Ambulance Scenario (C-xiv): The patient is in an ambulance. The sensors on the patient are connected to the ambulance's information system, which is connected to a hospital infrastructure via a mobile network connection. The communication between the patient's sensors is either directly to the ambulance infrastructure, or via the mobile phone. The ambulance and the patient's mobile phone may use different carriers. Some properties in this scenario are common with Scenario iv (vehicle transport).

Note that once the patient is inside the ambulance, sensors should communicate with devices in the ambulance without involving the mobile carrier.

IV. Storylines for the Scenarios

The set of overall scenarios, core scenarios, and transitions can be used to create *storylines* that can be used as case studies in ASSET. We present the storylines developed for the Scenarios A and B. Parts of these storylines will be used in the following analysis to evaluate the diverse functions in the IoT. We have not yet developed a storyline for Scenario C.

A. Storyline for the Home Scenario

We developed the storyline for the home scenario as follows: Petra has both a heart condition and diabetes. In a hospital, she had two sensors placed in or on her body: one heart sensor and one blood sugar sensor. In addition, she uses external sensors to measure blood pressure, heart beat, inertial sensors, etc., as well as a camera. Inertial sensors can be used to detect if Petra falls in order to automatically call for help while cameras could be used to assess her mood [20]. Petra is living in her home that has been prepared for the monitoring system and is commissioned with the necessary data connections so that her vital signs can be periodically reported to the healthcare personnel in levels (II) (nurse or doctor) or (III) (patient records) as introduced in Fig. 3; several technologies can be applied to achieve this.

The patient monitoring system is set up so that the sensor data are transmitted wirelessly (several transmission technologies are possible) to a smartphone that acts as PCH. The PCH communicates with the hospital infrastructure (Level (III)).

1. Petra is now being monitored at home but data are acquired remotely (A-i); the following requirements are important:
   a. Petra wants the data related to her medical condition to remain confidential from neighbours, i.e., people close-by, but outside her home. The confidentiality requirement includes physiological data, location data, data retrieved from a smart-home environment, such as temperature and humidity, as well as other metadata and health records.
   b. Petra wants her data to remain confidential from visitors, i.e., people inside her home.
2. Petra takes a bath in her home (planned sensor acquisition disruption; A-i);
   a. the sensors are water-proof; the PCH is close enough to receive signals;
   b. the sensors need to be removed;
      i. a change in the values implicitly indicates the sensor removal; or
      ii. patient must notify the PCH about the sensors going off-line;
3. Petra is sleeping and sensors fall off (unplanned sensor acquisition disruption; A-i).
4. Petra leaves her home for training outdoors or a stroll in the park nearby (A-ii);
   a. she is walking alone with her sensors communicating to the PCH;
   b. she meets an acquaintance, Linda who has similar sensor equipment; note that Petra's sensors could communicate through Linda's sensor network; they continue walking together;
   c. when they walk further, Petra looses the communication channel to the health care institution because of the terrain. She could either connect through the open, mobile WLAN-zones that are offered or use Linda's PCH as communication channel.
5. Petra leaves her home to visit her friends in a café (A-vi, A-ii, A-iii, A-iv).
6. Petra visits her regular doctor for a check-up; the doctor's office is within walking distance from her home (A-ii, A-vii, A-viii).
7. Petra becomes ill and is transported by an emergency ambulance to the hospital (B-xiv); transition to the Overall Hospital Scenario B.

B. Storyline for the Hospital Scenario

We developed the storyline for the hospital scenario as follows: Petra has both a heart condition and diabetes. One year ago, she had two sensors placed in or on her body: one heart sensor and one blood sugar sensor that both communicate wirelessly. In addition, she uses external sensors as described for the storyline of Scenario A. Petra suddenly gets ill while being at home. This is detected by the patient monitoring system installed at her home.

1. Petra is taken in an ambulance to the hospital (B-xiv). In addition to the sensors she is using, the paramedics use EEG and ECG sensors. The information from all sensors is available in the ambulance from three possible sources:
   a. information received directly from the sensors, available on the displays in the ambulance;
   b. information received from the PCH that Petra is using;
   c. information received from the healthcare records.
2. After the ambulance arrives at the hospital, Petra is moved to a room where diagnosis of her condition is performed (B-ix). Different sensors are used to find out her condition. These sensors are removed after diagnosis.
3. It becomes clear that Petra needs to undergo surgery (B-x). During surgery sensors are used to measure certain biomedical values. However, the medical procedure also creates electromagnetic noise in the same band as the data transmission between sensors uses.
4. After the surgery, Petra is moved to intensive care (B-xi) where a variety of sensors are used to observe her biomedical values.
5. After two days, Petra is moved to a recovery room with three other patients to allow time for her surgery wound to heal and for observation (B-xii). In addition to the heart and blood sugar sensors, two additional sensors are now used, but these will be removed after the observation phase is over. The two other patients in the same room are using the same kind of sensors.
   a. The sensors Petra is using transmit their readings to her PCH.
   b. Petra's additional sensors transmit their readings to a base station in the patients' room, while her ordinary sensors are still report to her PCH.
6. Petra is discharged from hospital; transition to Overall Scenario A.

C. Applying the Storylines

As described by Savola and Abie [21] and Savola et al. [19] the data integrity, privacy, data confidentiality, availability, and non-repudiation requirements should be met for all core scenarios and communication levels presented in Section III, specifically end-user authentication and authorisation for scenarios in Levels (0)-(II), sensor and WBSN authentication for scenario in Level (I), service provider user authentication for scenarios in Levels (III) and (IV), and service provider user authorisation in Levels (III) and (IV). This is also true for both storylines described above since these scenarios apply to both storylines but in varying situations and contexts. The adaptive security requirements for both storylines therefore can be summarised as follows:

1) End-user authentication and authorisation: The adaptive authentication mechanisms must cope with changing context of use, security threats and the user behaviour in order to enforce context-aware authentication mechanisms in an efficient and usable manner.

2) Sensor and WBSN authentication: Adaptive authentication mechanisms must cope with critical decisions to be made by the end-user and the service provider user based on the sensor input in order to minimise the possibility of fake sensors in possibly varying situations.

3) Service provider user authentication: Adaptive authentication mechanisms must cope with changing demands depending on the privacy level and the official authorisation level for making treatment decisions.

4) Service provider user authorisation: Adaptive authorisation techniques must cope with setting the adequate requirements and enforcing the sufficient authorisation mechanisms based on the strength of the authentication, context, and user role.

5) Data integrity (all levels): Adaptive data integrity techniques must maintain adequate data integrity especially during alarm situations allowing patients health security and longer-time treatment decisions

6) Privacy and data confidentiality (all levels): Adaptive security decision-making must adapt to privacy and data confidentiality requirements based on the data processing needs, roles of stakeholders, regulations and legislation, and the privacy level of data indicated by privacy metrics. Since context can affect privacy, adaptive security must be able to adapt to different types of context such as time, space, physiological parameter sensing, environmental sensing, and noisy data. The context must also be collected and evaluated in real time in a secure and accurate manner.

7) Availability (all levels): Adaptive techniques must balance the load in the system and use resilience solutions to maintain adequate availability, which is critical for health and life.

8) Non-repudiation (all levels): Adaptive authentication mechanisms must ensure the adequate non-repudiation level despite of changing conditions and selection of security controls.

Walking through these story lines or threat analysing them will show that the above adaptive security requirements must be met for their success and proper functioning. For example, the security requirement pointed out in Step 1.a of the storyline is related to confidentiality and privacy, which are often emphasised in healthcare. Strong confidentiality algorithms, key distribution, associated processes, and compliance to appropriate privacy legislation and regulations are crucial.

V. Evaluating the Home Scenario

We use selected parts of Scenario A to illustrate how to use the ASSET framework. We go through the scenario

TABLE II: Numeric results for Example 1: applying the ASSET framework using the logarithmic approach from eq. (3)

	S_1		S_2		S_3		S_4		Q_1		Q_2		C		q_{total}
w_i	0.4		0.8		2.0		1.0		1.0		1.5		1.0		$\sum = 7.4$
	$\tilde{q}$	$\hat{q}$	$\tilde{q}$	$\hat{q}$	$\tilde{q}$	$\hat{q}$	$\tilde{q}$	$\hat{q}$	$\tilde{q}$	$\hat{q}$	$\tilde{q}$	$\hat{q}$	$\tilde{q}$	$\hat{q}$	q_{total}
Case I	6	0.997	8	0.991	8	0.977	10	1.000	10	1.000	10	1.000	10	1.000	0.965
Case II	6	0.997	10	1.000	8	0.977	10	1.000	10	1.000	10	1.000	1	0.846	0.824
Case III	7	0.998	9	0.996	8	0.977	9	0.995	8	0.988	10	1.000	10	1.000	0.954
Case IV	6	0.997	8	0.991	10	1.000	10	1.000	8	0.988	9	0.992	10	1.000	0.968
Case V	6	0.997	8	0.991	9	0.989	10	1.000	9	0.995	10	1.000	10	1.000	0.972
Case III+IV	6	0.997	9	0.996	10	1.000	10	1.000	9	0.995	10	1.000	10	1.000	0.987

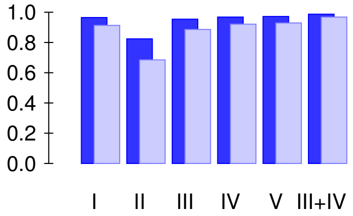


Fig. 7: Visualising the results for q_{total} from Example 1. The dark blue bars represent the results using the logarithmic function e_β, as shown in eq. (3), while the light blue bars represent the results using the linear function e_α, as shown in eq. (2).

TABLE III: The 11-value scale for $\tilde{q}_{S_2}$ of Example 1

$\tilde{q}_{S_2}$	Description
10	not observable outside apartment
9	barely observable in adjacent apartments; cannot be interpreted
8	barely observable in adjacent apartments; need advanced equipment to interpret
7	observable in parts of adjacent apartments, but not beyond
6	well observable in adjacent apartments, but not beyond
5	observable in range > 30m; on street
4	observable in range > 50m on street
3	observable in range > 100m on street
2	observable on street from running car
1	observable through wide-range network
0	n/a

description, and comment on the use of the framework. Note, however, that the numerical values are for illustration purposes. These values are based on rough estimates instead of a careful assessment. Different methods for assessment were proposed above in Section II-C, but applying and evaluating the different methods remain future work.

A. Confidentiality and Observability

In the storyline of the Home Scenario, Petra is monitored at home with the requirement that she wants her data to be confidential for people inside and outside her home. Let us assume that the properties of data observability and data confidentiality are essential in this first case, i.e., are in $\mathcal{S}$.

Here, data observability means that a third party can observe the signal sent from a device and, thus, deduce the existence of this device and some meta-data. For instance, neighbours of Petra may observe the signals from her sensors and make assumptions about her health conditions from this. As countermeasures the apartment could be shielded or the signal strength of the sensors could be reduced. While shielding the apartment is too expensive, reducing the signal strength, however, could have an impact on the data availability since some corners in Petra's apartment would not be covered.

Data confidentiality means that a third party cannot interpret the received signals. Cryptographic methods and authentication are often used to assure data confidentiality. Countermeasures when threats occur could use a different cryptographic method or authentication protocol. However, using a different cryptographic method could have a negative impact on the performance or battery consumption.

For a numeric example, here denoted as Example 1, we use the following variables: q_{S_1} is the value for observability inside the apartment; q_{S_2} is the value for observability outside the apartment; q_{S_3} is the value for confidentiality; q_{S_4} is the value for availability; q_{Q_1} is the value for bandwidth; q_{Q_2} is the value for battery consumption; and q_C are other mitigation costs. Recall that the value of q_i indicates how far a given requirement is from the ideal fulfilment, where 1 is complete fulfilment of the requirement. We use the following cases: *I*) the base case, i.e., the apartment is not shielded, rather simple encryption algorithms and authentication protocols are used, and sensors transmit at normal power; *II*) shielding the apartment; *III*) reducing transmission power; *IV*) using different encryption algorithm; and *V*) using different authentication protocol.

As outlined in in Section II-B, for objective assessment we need to establish a scale using n steps similarly to the Likert scale. For an example, we present a possible scale for the requirement $\tilde{q}_{S_2}$ (observability outside apartment) on a scale with 11 values in TABLE III. The value of $\tilde{q}_{S_2} = 0$ is marked as not applicable to indicate that for observability outside the apartment no situation is considered totally unacceptable. Note that marking $\tilde{q}_{S_2} = 0$ implies $q = 0$ for this alternative, i.e., it would be marked as totally unacceptable.

In an experiment, we assessed the values for $\tilde{q}_{S_{i=1\ldots4}}$, $\tilde{q}_{Q_{i=1\ldots2}}$, and $\tilde{q}_{Q_C}$ by using a rough estimate. We also assigned values for the weights w_i using intuition; we are aware that these values need to be assessed more thoroughly at a later stage. The assessment values, weights, and results for $\hat{q}_i$ and q_{total} are shown in TABLE II for the logarithmic approach from eq. (3). We also applied the linear approach from eq. (2) to the same data. Both results for q_{total} are visualised in Fig. 7.

In our example we see that the logarithmic approach and the linear approach show similar behaviour with respect to ranking

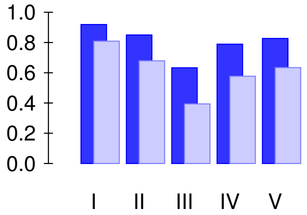


Fig. 8: Visualising the results from Example 2. The dark blue bars represent the results using the logarithmic function e_β, as shown in eq. (3) while the light blue bars represent the results using the linear function e_α, as shown in eq. (2).

TABLE IV: Example 2 for applying the ASSET framework using the logarithmic approach from eq. (3)

q_i	S_1	S_2	S_3	S_4	Q_1	Q_2	C	q_{total}
w_i	1	1	2	1	1	1.5	1	$\sum = 8.5$
Case I	7	6	8	9	8	9	10	0.919
Case II	7	6	4	9	8	9	9	0.850
Case III	7	5	4	1	1	8	9	0.633
Case IV	7	5	3	9	7	7	8	0.789
Case V	7	6	4	9	6	8	8	0.827

the alternatives. However, the logarithmic approach results in higher values and less differences for the values in-between. In this particular example, a combination of cases III and IV, gives the best result while case II delivers the lowest result, which is reasonable.

B. Assessment of Changes in Time

As Example 2 we use the part of the storyline where Petra is taking a stroll in the park. We assume that her sensors are connected wirelessly to her smartphone in its function as a PCH, and the PCH is communicating through a wireless network with the health care infrastructure through a public wireless network offered by a telephony provider. Further, we assume that her smartphone can connect using a WLAN.

In this example, we use different definitions for q_{S_1} and q_{S_1} by using the observability of the sensors and the PCH, respectively. We take into account effects for wide area networks that indicate that battery consumption is higher when the signal strength from the base station is weak or the connection is lost.

For a numeric example we use the following variables: q_{S_1} is the value for observability of the sensors; q_{S_2} is the value for observability of the PCH; q_{S_3} is the value for confidentiality; q_{S_4} is the value for availability; q_{Q_1} is the value for bandwidth; q_{Q_2} is the value for battery consumption; and q_C are other mitigation costs. We use the following cases from the storyline of Scenario A: *I*) walking alone in the park; *II*) meeting Linda; *III*) loosing connection; *IV*) connect to open, mobile WLAN; and *V*) using Linda's PCH as communication channel.

In an experiment, as above, we assessed the values for $\tilde{q}_{S_{i=1\ldots4}}$, $\tilde{q}_{Q_{i=1\ldots2}}$, and $\tilde{q}_C$ by using a rough estimate and assigned values for the weights w_i using intuition. The assessment values $\tilde{q}_i$, weights, and q_{total} are shown in TABLE IV for the logarithmic approach from eq. (3). We also applied the linear approach from eq. (2) to the same data. Both results for q_{total} are visualised in Fig. 8.

In this example we see how the security situation changes due to changes of the context (I–II–III), i.e., when Petra meets Linda or Petra looses connection. This example also shows that the assessment can give a hint which one of two possible actions (IV or V) would promise a better security situation.

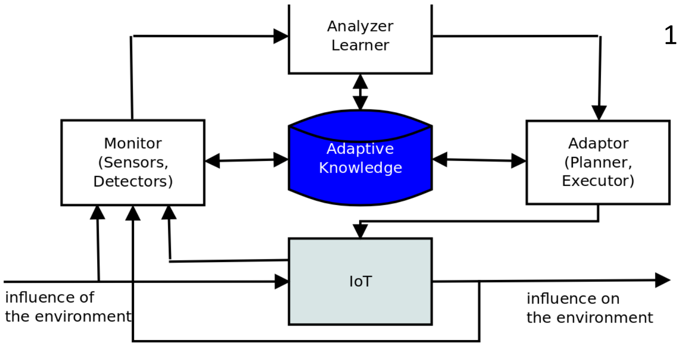


Fig. 9: The Adaptive Security concept, adapted for the IoT by Abie [24].

VI. Applying the Framework to Adaptive Security

Abie and Balasingham [2] define the term *adaptive security* as *"a security solution that learns, and adapts to changing environment dynamically, and anticipates unknown threats without sacrificing too much of the efficiency, flexibility, reliability, and security of the IoT system".* Abie and Balasingham present the *Adaptive Risk Management* (ARM) framework that is based on a feedback loop known from cybernetics [22] with the five measures (*i*) identify, (*ii*) analyse, (*iii*) plan, (*iv*) track, and (*v*) control. This results in four steps in the adaptation loop, aligned to ISO/IEC 27005:2008 [10] and the *Plan–Do–Check–Act* (PDCA) model of ISO/IEC 27001:2005 [23].

Abie [24] presented a functional description on the concept of adaptive security for a message-oriented infrastructure; he adapted this concept to the IoT, as shown in Fig. 9. He identified the following functionality to be essential for adaptive security to be implemented: *a*) being self-aware using a feedback loop and a history database; *b*) being context-aware using sensors and feedback from other nodes in the IoT; *c*) using security metrics to process the data from the sensors and the other nodes; *d*) using risk and threat estimation and prediction; *e*) using security metrics as defined by Savola et al. [19]; *f*) using methods such as Bayesian networks [25], game theory, Markov chains, etc. to support the threat estimation and prediction; *g*) using a decision making module to enforce appropriate security and privacy level; and *h*) communicating data to other nodes in the IoT.

A. Integrating the Estimation Function to Adaptive Security

In the adaptive security concept, the Monitor receives data from sensors, detectors, and other sources that are further used in the Analyser/Learner to make adaptive decisions. In this context, the ASSET evaluation framework can be used to

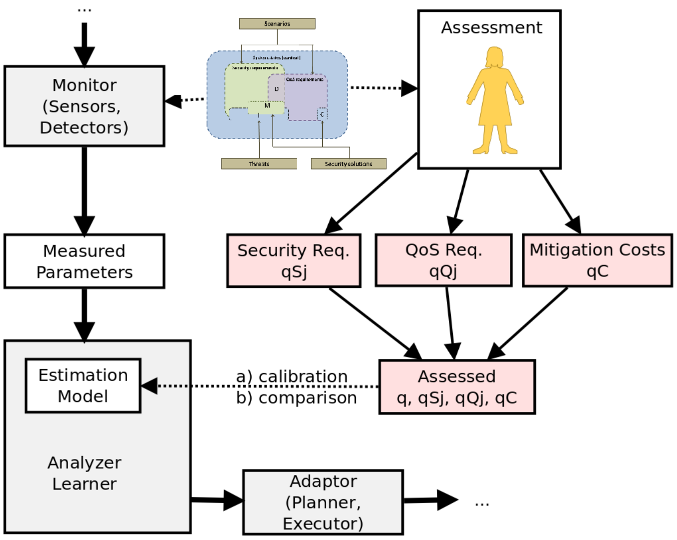


Fig. 10: Integration of the estimation framework into the adaptive security model.

provide the ground truth data *a*) to train the learning algorithms employed in the evaluation loop, and *b*) to evaluate whether the behaviour of the adaptive algorithms is reasonable.

For this we follow the following recipe: We use the storylines similarly as done in Section V where we assess the values q_i for all useful cases that can appear and calculate q with the suitable weights. On the other hand, the Monitor receives k measured values from diverse sensors and detectors. These measurements are denoted as s_k.

We postulate a function $u(s_k)$ that ideally is designed such that $u(s_k) = q$ for q as defined in eq. (1) for all relevant situations from the scenarios. In that way, using the function u, the input from the sensors and detectors will generate the same value as the assessment suggests. Alternatively, we can postulate functions $u_i(s_k)$ where $u_i(s_k) = q_i$ for all relevant situations. It is intended that the function u_i will generate the same value for each partial product as the assessment suggests.

The functions $u(s_k)$ and $u_i(s_k)$ could be instantiated in a learning phase. However, the adaptive security model can also handle this dynamically, so that the definition of these functions can vary over time.

The evaluation of the functions $u(s_k)$, respectively $u_i(s_k)$, will be handled in the Analyser and Learner components using regression, Bayesian networks, game theory, or similar. On the basis of the evaluated values from these functions, the Adaptor takes the necessary decisions. Fig. 10 illustrates how the estimation function can be integrated into the adaptive security model.

Based on the sets of system states the assessment of the values q, q_{S_i}, q_{Q_i}, and q_C is performed using a panel of users and specialists. This is shown on the right side of Fig. 10. These values are used to calibrate the estimation model. When the estimation function is established, these values can also be used to be compared with the estimation function for validation purposes. On the left side of Fig. 10, a part of the adaptation loop is shown with the components Monitor, Analyzer, and Adaptor. The Monitor component retrieves data from sensors and adaptors to a set of measured parameters. Using the estimation model, the Analyzer performs its tasks, and forwards the calculated values to the Adaptor component.

B. Evaluation Methods

For the purpose of evaluating the behaviour of the adaptive security methods we intend to employ the scenarios and storylines presented in Section V above together with implementations in a lab [26], simulation, and formal reasoning [27]. In an evaluation, we will go through each situation of the storylines, and assess or calculate the values q, q_{S_i}, q_{Q_i}, q_C, $u(s_k)$, and $u_i(s_k)$ as necessary.

Using a lab one could build all necessary equipment that contains all necessary functionality. According to the evaluation method, one will go through all states and situations defined by the storyline and assess or calculate all relevant values according to our framework. Thereafter, the adaptation algorithm will be applied, resulting in new states that are evaluated by assessing and calculating the relevant values. Comparing the calculated values $u(s_k)$, and $u_i(s_k)$ after each adaptation step with the desired and assessed values for q, q_{S_i}, q_{Q_i}, and q_C will give evidence on the behaviour of the adaptation algorithm. The goal is to evaluate whether the behaviour of the adaptation loop is close to the “right” decisions deduced from the assessment.

Note that in the absence of a lab, simulations or the use of formal methods can be considered. Here, instead of implementing the devices in real hardware the essential functionality is implemented in a model, and simulation and model checking techniques are used [27], [28].

VII. Discussion and Related Work

Our framework supports the evaluation of security solutions and provides a means to assess data for development and calibration of the estimation model. In this section, we discuss several issues regarding our framework. We also relate our work to frameworks that are described in the literature.

A. Issues and Concerns

The estimation model’s design and the function $u(s_k)$ are not the focus here, but our framework can calibrate an estimation model. In principle, methods from machine learning, such as regression analysis, Bayesian networks, fuzzy logic, or game theory can be used to develop the function $u(s_k)$ introduced in Section VI. The assessed data will be used as training data while only the measurable data from sensors will be used in the adaptive security concept.

Using this concept, the estimation model and the function $u(s_k)$ will respond with a sufficiently correct estimate as long as the particular case has been part of the scenarios and storylines used in the assessment. Cases that are not covered in the assessment can still be estimated, but we cannot predict the appropriateness of the estimate. Thus, the framework needs to monitor continuously whether all relevant cases are covered, and refine the estimation model with new assessments when missing cases are discovered.

In contrast to the adjustment of the estimation model, the evaluation of the function $u(s_k)$ can be performed in near real-time. Depending on the estimation principle, the evaluation can be done using partial evaluation when only a few parameters are updated at a time. Metrics for evaluation of such estimation models can be found in the literature for the chosen estimation principle [29].

One concern of our assessment model is that we use human assessment, which introduces subjectivity into the assessment. While it is viable to check objectively whether a system state fulfils a catalogue of guidelines or requirements, the severity of deviations from the ideal state are subjective. For instance, in healthcare applications, patients or health personnel might need to make choices whether to accept deficiencies in privacy or security to use a service.

There are assessment methods that make use of evaluation panels consisting of both laymen and experts in other application areas. For example, the estimation of video and audio quality [11] can be performed using subjective evaluations with user panels where each panel member evaluates content under well-defined conditions. Another example is the evaluation of the quality of open source software [15] where methods using a user-based rating have been related to more objective methods. It has been shown that the subjective methods give an appropriate estimate of the software quality, despite their simplicity [16]. Hence, we argue that subjective assessment can be applied also for security, privacy, QoS, and costs.

In healthcare applications, the balance between security, privacy, and QoS needs to be addressed. For instance, non-critical information could be made available with lower security to a user or security could be lowered when emergency access to the medical data is necessary. Our assessment framework can address such issues by modelling these as cases into the scenarios and storylines. The evaluation will then show whether the q_{total} is within acceptable borders.

Similarly, our assessment method is also suitable to assess alternative security methods for specific target groups, such as people with disabilities. For instance, Scenario A could be extended by Petra using an alternative authentication method, e.g., the one described by Fuglerud and Dale [30].

B. Security Metrics

Security metrics [31] provide a comprehensive approach to measuring risk, threats, operational activities, and the effectiveness of data protection in software systems. They also provide the means to compare different solutions and obtain evidence about the performance of operational security in adaptive security [32]. Some often-used metrics include the number and frequency of vulnerabilities or actual attacks. These are based on appropriate security and privacy goals.

Savola and Abie [33] present a methodology for developing security metrics using a risk-driven process; this starts from an analysis of threat and vulnerability of the system and aims to achieve a collection of security metrics and related measurement architecture. This concept has been extended to include adaptive security management for healthcare applications [19].

Weiß et al. [34] propose security metrics built on a risk management approach. Jafari et al. [35] develop security metrics to assess security posture of healthcare organisations. Abie and Balasingham [2] integrate security metrics into the validation of risk-based adaptive security for deployment in the IoT under changing threat models. Our work complements these proposals with a scenario-based framework for the assessment and validation of context-aware adaptive security solutions. The security metrics described by Weiß et al. and Jafari et al. are well suited as objective data described in Section II-C3.

C. Security Evaluation Frameworks

Frameworks for evaluating security and privacy in eHealth include a Common Criteria framework for the evaluation of information technology systems security [36], a framework for information security evaluation [37], a requirement-centric framework for information security evaluation [38], a scenario-based framework for the security evaluation of software architecture [39], and the OWASP risk rating methodology [40].

Recently, Shoniregun et al. [41] proposed a unified security evaluation framework for healthcare information systems by exploring the solutions and technologies currently available for evaluating security and privacy problems in such systems. The authors acknowledged the limitations of nearly all major efforts to measure or assess security such as Trusted Computer System Evaluation Criteria (TCSEC), Information Technology Security Evaluation Criteria (ITSEC), Systems Security Engineering Capability Maturity Model (SSE-CMM), and Common Criteria. The authors also reviewed approaches to evaluation of healthcare information security and privacy, such as standards-based, privacy policy-based, ontology-based, security and privacy metrics-based, and model-based approaches to security and privacy evaluation.

Torjusen et al. [28] present a formal approach to verification of an adaptive security framework for the IoT, with integration of run-time verification enablers in the feedback adaptation loop and the instantiation of the resulting framework with Coloured Petri Nets for formally evaluating and validating self-adaptive security behaviour. Our work complements this concept by providing a scenario-based assessment to convey the design requirements.

D. eHealth Evaluation Frameworks

There are multiple evaluation frameworks for security and privacy in eHealth available, including the analysis of different parts, such as patient monitoring systems or electronic health record (EHR) systems. Note that an EHR can be part of the IoT in the sense that it is the data sink in a heath care organisation.

Fernández-Alemán et al. [42] conducted a comparative literature review of the security and privacy of the current EHR systems based on ISO 27799 [43]. They identified and analysed critical security and privacy aspects of EHR systems, such as the need for harmonisation of compliance standards to resolve possible inconsistencies and conflicts among them, the use of efficient encryption scheme for the acquisition, development and maintenance of information systems, access control, communication and operational management, and the security of human resources. They have also discovered that although most EHR systems defined security controls these arent fully deployed in actual tools. Note also that their emphasis is not on wireless communication which is often used in the IoT. The research framework by Fernández-Alemán et al. is based on a literature review that is static by nature, while our framework is able to assess values that change dynamically.

Malin et al. [44] described the problems, perspectives and recent advances in biomedical data privacy by illustrating the space of data privacy in the biomedical domain as multidisciplinary; it crosses ethical, legal, and technical boundaries. This demonstrates that appropriate socio-technical solutions can be defined for emerging biomedical systems that can balance privacy and data utility and system usability. At the same time this highlights cloud computing as new computing, high-throughput technology that creates new challenges to privacy that biomedical community will need to handle in the future. Malin et al.'s work addresses more policies in different domains than assessing the security and privacy characteristics.

Kierkegaard [45] highlights the benefits and the key concerns of a centralised supranational health network that allows access to health information anyplace and anytime by enhancing efficiency, effectiveness, accuracy, completeness and accessibility, and generally improving the quality of healthcare services. These benefits lead to an increase of the amount of information collected, processed, filtered, transferred or retained. In in turn, this increases the potential abuse and privacy threats to such information. Thus, privacy and data protection need to be embedded within the infrastructure. Note that a potential single point of failure of such an infrastructure is a major concern. Also Kierkegaard's work addresses more policies in different domains than assessing the security and privacy characteristics.

Boxwala et al. [46] proposed statistical and machine learning methods to help identify suspicious, i.e., potentially inappropriate, access to EHRs using logistic regression training and support vector machine models and concluded that such methods can play an important role in helping privacy officers detect suspicious access to EHRs. While their methods and ours can predict suspicious accesses (threats), our framework goes further by identifying a set of security solutions that mitigate these threats and a set of system states that represent the dynamic context in which the patient monitoring system operates and adapts specialised to the type of scenarios and story lines used.

Peleg et al. [47] presented a framework for situation-based access control for privacy management through modelling using object-process methodology to structure the scenarios and conceive a situation-based access control model. The framework is intended for traditional role-based access control. Their work and ours are similar in expressing scenarios of patients data access as a basis of preserving of patients security and privacy. They differ in that their solution is access control specific while ours is applicable to any security or quality of service requirements. The framework by Peleg et al. is qualitative while we have added quantitative components in our framework.

Note that the above described frameworks by Boxwala et al. and Peleg et al. can produce values that can be used as input values q_i for our framework, as described in Section II-C3.

VIII. Conclusion

We presented an evaluation framework for adaptive security to be applied for the IoT in eHealth applications. We highlighted the role of the scenarios in the evaluation framework. The framework is based on a generic system model, security and QoS requirements for eHealth applications, and a generic assessment framework. Further, the framework uses sets of states that are used to estimate how well the security and QoS requirements are fulfilled.

For evaluation purposes we presented three scenarios, a home scenario, a hospital scenario, and an emergency scenario. These scenarios are annotated with requirements and outlined as storylines which can be used to evaluate adaptive security algorithms. Our evaluation methodology is designed to compare results from lab experiments and simulations with the assessment by human observers.

The scenarios cover multiple core scenarios representing a range of eHealth IoT situations. These address specific requirements related to the context, the data-communication, the devices, and the actions of the involved actors. The core scenarios are specific to the eHealth case, and make it possible to identify relevant cases that need to be evaluated, such as situations where IoT devices need to be removed or disconnected, the use of ample communication channels, or the impact of mobility.

Storylines for a patient with chronic diseases have been described and analysed. In the future, the overall scenarios, as well as the underlying core scenarios and storylines will be used in the ASSET project to evaluate the adaptive security algorithms. We posit that the framework, methodologies, and scenarios presented here can be used as a blueprint for evaluations of adaptive algorithms beyond the analysis of the adaptive algorithms of the ASSET project.

IX. Acknowledgments

The work presented here has been carried out in the project ASSET – Adaptive Security for Smart Internet of Things in eHealth (2012–2015) funded by the Research Council of Norway in the VERDIKT programme. We wish to thank our colleagues involved in this project for helpful discussions that made this study possible. Particularly, we want to thank Ragnar Hauge for discussions while developing this work. We also wish to thank the anonymous reviewers for valuable comments.

References

[1] W. Leister, M. Hamdi, H. Abie, and S. Poslad, "An evaluation scenario for adaptive security in eHealth," in PESARO 2014 – The Fourth International Conference on Performance, Safety and Robustness in Complex Systems and Applications. IARIA, 2014, pp. 6–11.

[2] H. Abie and I. Balasingham, "Risk-based adaptive security for smart IoT in eHealth," in BODYNETS 2012 – 7th International Conference on Body Area Networks. ACM, 2012.

[3] I. F. Alexander and N. Maiden, Eds., "Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle". John Wiley & Sons, 2004.

[4] S. Faily and I. Flechais, "A meta-model for usable secure requirements engineering," in SESS – ICSE Workshop on Software Engineering for Secure Systems. Association for Computing Machinery (ACM), 2010.

[5] H. Mouratidis and P. Giorgini, "Security attack testing (SAT)–testing the security of information systems at design time," Information Systems, vol. 32, no. 1, Jan. 2007, pp. 1166–1183.

[6] ITU-T, "Overview of the Internet of Things," International Telecommunication Union, Recommendation Y.2060 (06/2012), 2013. [Online]. Available: http://www.itu.int/ITU-T/recommendations/rec.aspx?rec=11559 [Accessed: 13. Nov. 2014].

[7] O. Vermesan, P. Friess, P. Guillemin, H. Sundmaeker, M. Eisenhauer, K. Moessner, F. L. Gall, and P. Cousin, "Internet of things strategic research and innovation agenda," in Internet of Things–Global Technological and Societal Trends. River Publishers, 2011, pp. 7–151.

[8] K. Habib and W. Leister, "Adaptive security for the Internet of Things reference model," in Proceeding of Norwegian Information Security Conference, NISK 2013, C. Rong and V. Oleshchuk, Eds., 2013, pp. 13–24.

[9] W. Leister, H. Abie, and S. Poslad, "Defining the ASSET scenarios," Norsk Regnesentral, NR Note DART/17/2012, Dec. 2012.

[10] "ISO/IEC 27005:2008 Information technology–Security techniques–Information security risk management," International Organization for Standardization and International Electrotechnical Commission, standard, 2008.

[11] W. Leister, S. Boudko, and T. H. Røssvoll, "Adaptive video streaming through estimation of subjective video quality," International Journal on Advances in Systems and Measurements, vol. 4, no. 1&2, 2011, pp. 109–121. [Online]. Available: http://www.iariajournals.org/systems_and_measurements/ [Accessed: 1. Nov 2014].

[12] R. Likert, "A technique for the measurement of attitudes." Archives of Psychology, vol. 22, no. 140, 1932, pp. 1–55.

[13] A. Nevill and A. Lane, "Why self-report likert scale data should not be log-transformed," Journal of Sports Sciences, vol. 25, no. 1, 2007, pp. 1–2.

[14] J. Perl and S. Russell, "Bayesian networks," in Handbook of Brain Theory and Neural Networks, M. Arbib, Ed. Cambridge, MA: MIT Press, 2003, pp. 157–160.

[15] A. Wasserman, M. Pal, and C. Chan, "The business readiness rating model: an evaluation framework for open source," in Proc. EFOSS Workshop, Como, Italy, Jun. 2006.

[16] A.-K. Groven, K. Haaland, R. Glott, and A. Tannenberg, "Security measurements within the framework of quality assessment models for free/libre open source software," in Proc. Fourth European Conference on Software Architecture: Companion Volume, ser. ECSA '10. New York, NY, USA: ACM, 2010, pp. 229–235.

[17] W. Leister, T. Fretland, and I. Balasingham, "Security and authentication architecture using MPEG-21 for wireless patient monitoring systems," International Journal on Advances in Security, vol. 2, no. 1, 2009, pp. 16–29. [Online]. Available: http://www.iariajournals.org/security/ [Accessed: 1. Nov 2014].

[18] W. Leister, T. Schulz, A. Lie, K. H. Grythe, and I. Balasingham, "Quality of service, adaptation, and security provisioning in wireless patient monitoring systems," in Biomedical Engineering Trends in electronics, communications and software. INTECH, 2011, pp. 711–736.

[19] R. M. Savola, H. Abie, and M. Sihvonen, "Towards metrics-driven adaptive security management in e-health IoT applications," in BODYNETS 2012 – 7th International Conference on Body Area Networks. ACM, 2012.

[20] J. Hernandez, M. E. Hoque, and R. W. Picard, "Mood meter: large-scale and long-term smile monitoring system," in ACM SIGGRAPH 2012 Emerging Technologies, ser. SIGGRAPH '12. New York, NY, USA: ACM, 2012, pp. 15:1–15:1.

[21] R. M. Savola and H. Abie, "Metrics-driven security objective decomposition for an e-health application with adaptive security management," in ASPI 2013 – International Workshop on Adaptive Security & Privacy management for the Internet of Things. ACM, 2013.

[22] W. R. Ashby, "An Introduction to Cybernetics". London: Chapman & Hall, 1957.

[23] ISO, "ISO/IEC 27001:2005 Information technology – Security techniques – Information security management systems – Requirements," International Organization for Standardization and International Electrotechnical Commission, standard, 2005.

[24] H. Abie, "Adaptive security and trust management for autonomic message-oriented middleware," in IEEE Symposium on Trust, Security and Privacy for Pervasive Applications (TSP 2009). Macau, China: IEEE, 2009, pp. 810–817.

[25] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Representation and Reasoning Series". Morgan Kaufmann, 1988.

[26] Y. B. Woldegeorgis, H. Abie, and M. Hamdi, "A testbed for adaptive security for IoT in eHealth," in ASPI 2013 – International Workshop on Adaptive Security & Privacy management for the Internet of Things. ACM, 2013.

[27] W. Leister, J. Bjørk, R. Schlatte, E. B. Johnsen, and A. Griesmayer, "Exploiting model variability in ABS to verify distributed algorithms," International Journal On Advances in Telecommunications, vol. 5, no. 1&2, 2012, pp. 55–68. [Online]. Available: http://www.iariajournals.org/telecommunications/ [Accessed: 1. Nov 2014].

[28] A. B. Torjusen, H. Abie, E. Paintsil, D. Trcek, and A. Skomedal, "Towards run-time verification of adaptive security for IoT in eHealth," in Proc. 2014 European Conf. on Software Architecture Workshops, ser. ECSAW '14. New York, NY, USA: ACM, 2014, pp. 4:1–4:8.

[29] B. G. Marcot, "Metrics for evaluating performance and uncertainty of Bayesian network models," Ecological Modelling, vol. 230, 2012, pp. 50–62.

[30] K. S. Fuglerud and Ø. Dale, "Secure and inclusive authentication with a talking mobile one-time-password client," IEEE Security and Privacy, vol. 9, no. 2, 2011, pp. 27–34.

[31] S. C. Payne, "A guide to security metrics," SANS Institute Information Security Reading Room, whitepaper, June 2006.

[32] A. Evesti, H. Abie, and R. M. Savola, "Security measuring for self-adaptive security," in Proc. 2014 European Conf. on Software Architecture Workshops, ser. ECSAW '14. New York, NY, USA: ACM, 2014, pp. 5:1–5:7.

[33] R. M. Savola and H. Abie, "Development of measurable security for a distributed messaging system," Intl. Journal on Advances in Security, vol. 2, no. 4, 2009, pp. 358–380. [Online]. Available: http://www.iariajournals.org/security/ [Accessed: 1. Nov 2014].

[34] S. Weiß, O. Weissmann, and F. Dressler, "A comprehensive and comparative metric for information security," in Proc. IFIP Intl. Conf. on Telecommunication Systems, Modeling, and Analysis 2005 (ICTSM2005), Dallas, TX, USA, 2005, pp. 1–10.

[35] S. Jafari, F. Mtenzi, R. Fitzpatrick, and B. O'Shea, "Security metrics for e-Healthcare information systems: A domain specific metrics approach," Intl. Journal of Digital Society (IJDS), vol. 1, no. 4, 2010, pp. 238–245.

[36] R. Kruger and J. H. P. Eloff, "A common criteria framework for the evaluation of information technology systems security," in Information Security in Research and Business, Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC '97), 14-16 May 1997, Copenhagen, Denmark, ser. IFIP Conference Proceedings, L. Yngström and J. Carlsen, Eds., vol. 92. Chapman & Hall, 1997, pp. 197–209.

[37] R. von Solms, H. van de Haar, S. H. von Solms, and W. J. Caelli, "A framework for information security evaluation." Information & Management, vol. 26, no. 3, 1994, pp. 143–153.

[38] R. Savola, "A requirement centric framework for information security evaluation," in Advances in Information and Computer Security, First International Workshop on Security, IWSEC 2006, Kyoto, Japan, October 23-24, 2006, Proceedings, ser. Lecture Notes in Computer Science, H. Yoshiura, K. Sakurai, K. Rannenberg, Y. Murayama, and S. Kawamura, Eds., vol. 4266. Springer, 2006, pp. 48–59.

[39] A. Alkussayer and W. Allen, "A scenario-based framework for the security evaluation of software architecture," in 3rd IEEE Intl. Conf. on Computer Science and Information Technology (ICCSIT), vol. 5, July 2010, pp. 687–695.

[40] "The OWASP risk rating methodology." [Online]. Available: https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology [Accessed: 17 November 2014].

[41] C. Shoniregun, K. Dube, and F. Mtenzi, "Towards a unified security evaluation framework for e-healthcare information systems," in Electronic Healthcare Information Security, ser. Advances in Information Security. Springer US, 2010, vol. 53, pp. 151–172.

[42] J. L. Fernández-Alemán, I. C. Señor, P. Á. O. Lozoya, and A. Toval, "Security and privacy in electronic health records: A systematic literature review," Journal of Biomedical Informatics, vol. 46, no. 3, 2013, pp. 541–562.

[43] "ISO/IEC 27799:2008. Health Informatics–Information security management in health using ISO/IEC 27002". Geneva, Switzerland: International Organization for Standardization, 2008.

[44] B. A. Malin, K. E. Emam, and C. M. O'Keefe, "Biomedical data privacy: problems, perspectives, and recent advances," JAMIA, vol. 20, no. 1, 2013, pp. 2–6.

[45] P. Kierkegaard, "Electronic health record: Wiring europes healthcare," Computer Law & Security Review, vol. 27, no. 5, 2011, pp. 503 – 515.

[46] A. A. Boxwala, J. Kim, J. M. Grillo, and L. Ohno-Machado, "Using statistical and machine learning to help institutions detect suspicious access to electronic health records," JAMIA, vol. 18, no. 4, 2011, pp. 498–505.

[47] M. Peleg, D. Beimel, D. Dori, and Y. Denekamp, "Situation-based access control: Privacy management via modeling of patient data access scenarios," Journal of Biomedical Informatics, vol. 41, no. 6, 2008, pp. 1028–1040.

[48] M. Peleg, S. Keren, and Y. Denekamp, "Mapping computerized clinical guidelines to electronic medical records: Knowledge-data ontological mapper (KDOM)," Journal of Biomedical Informatics, vol. 41, no. 1, 2008, pp. 180–201.

Respecting User Privacy in Mobiles: Privacy by Design Permission System for Mobile Applications

Karina Sokolova*†, Marc Lemercier*

*University of Technology of Troyes
Troyes, France
{karina.sokolova, marc.lemercier}@utt.fr

Jean-Baptiste Boisseau†

†EUTECH SSII
La Chapelle Sain Luc, France
{k.sokolova, jb.boisseau}@eutech-ssii.com

Abstract—**The Privacy by Design concept proposes to integrate respect for user privacy into systems managing user data from the early design stage. This concept has increased in popularity and the European Union (EU) is enforcing it with a Data Protection Directive. Mobile applications have emerged onto the market and the current law and future directive are applicable to all mobile applications designed for EU users. It has so far been shown that mobile applications do not suit the Privacy by Design concept and lack transparency, consent and security. Current permission systems are judged as unclear for users. In this paper, we introduce a novel permission model suitable for mobile application that respects Privacy by Design. We show that such adapted permission system can improve not only the transparency and consent but also the security of mobile applications. Finally, we propose an example of the use of our system in mobile application.**

***Keywords*–permission, permission system, mobile, privacy by design, privacy, transparency, control, Android, iOS, application, development, software design, pattern, mobility, design, modelling, trust**

I. Introduction

The idea of this paper was first outlined in [1] and extended in this article.

Mobile devices continue to gain in popularity. Thousands of services and applications are proposed on mobile markets and downloaded every day. Smart devices have a high data flow, processing and storing large amounts of data including private and sensitive data. Most applications propose personalized services but simultaneously collect user data even without the user's awareness or consent. More and more users feel concerned about their privacy and care about the services they use. The survey conducted by TRUSTe in February 2011 shows that smartphone users are concerned about privacy even more than about security (second in the survey results) [2].

Nowadays, people are aware of the lack of privacy, especially while using new technological devices where information is collected, used and stored en masse (Big Data). Privacy regulation aiming to control personal data use is in place in many countries. European Union privacy regulation includes the European Data Protection Directive (Directive 95/46/EC) and the ePrivacy Directive. United States regulation includes the Children's Online Privacy Protection Act (COPPA) and the California Online Privacy Protection Act of 2003 (OPPA). Canada has the Personal Information Protection and Electronic Documents Act (PIPEDA) concerning privacy.

The Privacy by Design (PbD) notion proposes to integrate privacy from the system design stage [3] to build privacy-respecting systems. PbD proves systems can embed privacy without sacrificing either security or functionality. Some PbD concepts are already included in European data legislation; the notion is considered to be enforced in European Data Protection Regulation, therefore, systems made with PbD are compliant with the law. An application of PbD notions is not only a benefit for users but also a legal obligation for developers.

The PbD concept was first presented by Dr Ann Cavoukian. She proposes seven key principles of PbD enabling the development of privacy-respecting systems. The system should be proactive, not reactive, embed the privacy feature from the design stage, integrate Privacy by Default, respect user privacy, data use should be transparent to the end user and the user should have access to the mechanism of control of his data. Full functionality and end-to-end security should be preserved without any sacrifice [3].

Mobile privacy was discussed in 'Opinion 02/2013 on apps on smart devices' by the Article 29 Data Protection Working Party [4], in which the opinion on mobile privacy and some general recommendations were given. The article states that both the Data Protection Directive and the ePrivacy Directive are applicable to mobile systems and to all applications made for EU users. Data Protection Regulation is also applicable to mobile systems. The article defines four main problems of mobile privacy: lack of transparency, lack of consent, poor security and disregard for purpose limitation.

Many reports such as [5] propose recommendations on mobile privacy improvement repeating basic privacy notions (e.g., data minimization, clear notices) but the exact patterns or technical solutions are missing.

Permission systems are embedded in mobile systems and are a crucial part of mobile security and privacy. Nowadays, permission systems do not follow Privacy by Design notions. Many works concentrate on analysing and modelling the current permission systems [6][7][8], on improving current permission systems to give more control to the user [9][10][11][12] or to add additional transfer permissions [13], on visual representation of permissions [14], on user perceptions of current permission systems [15][16][17], on data flow analyses (possible data leakage detection) [18][19][20] and on current permission enforcement and verification [21][22][23][24].

Few works try to redefine permission systems. In [25], authors propose an ontology where the right is given to an actor to take action on data. Rules are defined using SWRL language. Complex rules forbidding or allowing data access under a particular condition can be added to the policy. It is not clear what other action than the 'access' is supported by the ontology. A firewall was implemented on Java and ported to Android. The perception of users and the applicability to real applications is not disguised.

Authors of [26] propose a per-data permission system: more fine grained than Android default permissions. Access permissions are granted by data stored in SQLite databases and the access can be restricted by a piece of data - column (only phone number, only names from contacts) or by group - raws (only work contacts can be accessed) and mixed - cell. Privacy policy is expressed with subject having (1) or not having (0) access to an object. The permission system was implemented on Android and included used contact list data access permissions.

Current works are often limited by the data they can manage (use only geolocation data, SMS, address book, etc.), number of actions (most include only the 'access' action). Most works modify the functionality of current permission systems: allow permission to be revoked, return fake or empty data to the functionality. The perception of users and the applicability to real applications is often not disguised.

To our knowledge no work has been conducted on redefining the permission system to fit the Privacy by Design notion or on adding the purpose to permissions.

The remainder of the paper is organized as follows: Section II describes current permission systems of iOS and Android and points out problems regarding Privacy by Design. Section III introduces our proposal: the pattern of the privacy-respecting permission system. We show that it can cope with the transparency, consent and purpose-disregard problems and also improve security. Section IV shows the application of our novel permission system to the real mobile application. The paper ends with a conclusion and future works.

II. Existing Mobile Permission Systems

In this section, we present current iOS and Android permission systems and evaluate those systems regarding the PbD notion. We take into account the full functionality allowed by the permission system, privacy by default, transparency and the control notions.

- Full functionality: possibility to use all functionalities available on the platform.
- Privacy by Default: the default configurations of the system are privacy protective.
- Transparency: user should clearly understand what data is used, how and for what purpose.
- Control: user should have full control over his personal data use.

We consider the privacy policy to be very important for the proactive and transparent system, therefore, we present the state of application privacy policy in both systems.

A. Permissions

iOS and Android have different strategies in regard to access to the device data. The iOS platform gives non-native applications access only to the functionalities listed in privacy settings: location services, contacts, calendar, reminder, photos, microphone and Bluetooth (sensitive data, such as SMS and e-mails are not shared at all). Recently, the connection to Facebook, Twitter, Flickr and Vimeo was added to the platform (iOS7). Full functionality is given up for privacy reasons as applications cannot use the full power of the platform but only a limited number of functionalities.

An iOS application should have permission to access the information listed above. By default, an installed application has no permission granted. The application displays a pop-up explaining what sensitive data it needs before accessing it. The user can accept or decline permission. If permission is declined, the corresponding action is not executed. If the permission is accepted, the application obtains access to the corresponding data. The user is asked to grant permission only once, but he can enable or disable such permission for each application in privacy settings integrated by default into the iOS. iOS thereby maintains transparency, control and privacy by default.

The Android system remains on the sharing principle. Full functionality is preserved: applications have access to all native applications' data and can expose the data themselves. Applications need permission to access the data, but unlike iOS, users should accept the full list of permissions before installing an application. While all permissions are granted, an application has full access to the related data. Some Android permissions tagged as 'dangerous' can be prompted to the user every time the data is going to be accessed, but it is rarely the case. Users see the list of dangerous permissions on the screen before installing the application.

Android proposes more than 100 default permissions and developers can add supplementary permissions. Multiple works show that users do not understand many default permissions and fail to judge the application privacy and security correctly using the full permission list [15][17]. Permissions do not clearly show what data is used for and how. Moreover, some other studies show the abusive usage of Android permissions by developers [27].

Some users do not check the Android permission list because they need a service and they know that all permissions should be accepted to obtain it. Android permission lists look like a license agreement on a desktop application that everybody accepts but very few actually read [28].

Android user does not have any control over permissions once the application is installed: permissions cannot be revoked. Android does not include an iOS-like system permission manager (privacy settings) by default, therefore, the user has to enable or disable the entire functionality to disable access to related data (e.g., Wi-Fi or 3G for Internet connection; GPS for geolocation) or to use additional privacy enhancing applications.

TABLE I
ANDROID AND IOS PERMISSION SYSTEMS COMPARISON

	Full Functionality	Default Settings	Transparency	Control
Android	+	-	-	-
iOS	-	+	-/+	+

Both iOS and Android default permission systems mostly inform about data access, but not about any other action that can be completed with the data. For example, no permission is needed to transfer the data. Android and iOS include permissions for functionalities that can be related to personal data transfer, such as Bluetooth and Internet. Permissions can be harmless to users, but there is no indication of whether personal data is involved in a transaction. This decreases the transparency of both platforms.

Android and iOS permissions do not include purpose explanation. An iOS application helps to understand the purpose by asking permission while in use, but if an application has a granted permission once for one functionality it could use it again for a different purpose without informing the user. Android users can only guess what permission is used for and whether the use is legitimate.

Table I shows the system differences regarding four main privacy notions: full functionality, transparency, control and privacy by default. One can see that the current Android permission system is lacking in transparency, control and default privacy; iOS sacrifices functionality and also lack of transparency. Permissions are often functionality-related and users fail to understand and to judge them. Personal data use is unclear and the purpose is missing.

B. Privacy Policy

Users should choose applications they can trust. Apple ensures that applications available on the market are potentially harmless, although Android users should judge the application for themselves with the help of information available on the market. The AppStore and Google Play provide similar information: name, description, screenshots, rating and user reviews.

The transparency and the proactivity of the system can be improved by including the privacy policy in the store. A user can be informed about the information collected and stored before he downloads the application. Without any privacy policy, the user can hardly evaluate the security and privacy of the application, only the functionality and stability of the system. In their feedback, users often evaluate the functionalities and user interfaces and report bugs, but they rarely indicate privacy and security problems.

iOS does not require developers to include the privacy policy in the application but only in applications directed at children younger than 13 years old. Apple encourages the use of privacy policy in the App Store Review Guidelines and iOS Developer Program License Agreement. Apple specifies that developers should ensure the application is compliant with all laws of the country the application is distributed in. On viewing the App Store Review Guidelines one can see that all Privacy by Design fundamental principles and data violation possibilities are covered by Apple verification. However, the exact evaluation process used by Apple remains secret and some privacy-intrusive applications may appear in the store. Until recently, Apple authorized the use of device identification. This identification number was not considered private. Many applications used this number to uniquely identify their users, therefore, many applications were considered privacy intrusive [29].

Google Play Terms of Service do not require any privacy policy to be added to the Android applications. Google provides an option to include the privacy policy but does not verify or enforce it. Google Developers Documentation provides recommendations and warns that the developer has a responsibility to ensure the application is compliant with the laws of the countries in which the application is distributed.

Some developers include a license agreement and privacy policy. According to [30] only 48% of the top 25 Android paid applications, 76% of the top 25 Android free applications, 64% of iOS paid applications and 84% of iOS free applications have included the privacy policy. Android includes the permission list in the store and this can be considered a privacy policy, but, as previously discussed, the list is unclear to the final user.

III. PRIVACY-RESPECTING PERMISSION SYSTEM

Mobile phones have significant data flow: information can be received, stored, accessed and sent by the application. Data can be entered by the user, retrieved from the system sensors or applications, come from another mobile application, arrive from servers or from other devices. Data can be shared on the phone with another application, with servers or other device.

The permission system is integrated into mobile operating systems; well designed, it makes a proactive privacy-respecting tool embedded in the system.

We propose to focus permissions on data and the action that can be carried out on this data, rather than on the technology used. The definition of purpose of the data use is included in our permission system.

Privacy Policy should be short and clear. Users should have a global vision of the data use and functionalities before they install an application. Users rarely read long involved policies, especially when they want a service and feel they have no choice but to accept all permissions. Our permission system enables a simple policy to be generated with a list of permissions.

A. Permission definition

We model our permission system with an access control model. We choose discretionary access control where only data owner can grant access. The user should be able to control the data, therefore, we consider the user is a unique owner of all information related to him.

R_{app} is a set of $rules$ assigned to the application. We define a $rule$ as an assignment of the $\mathcal{R}ight$ over an $\mathcal{O}bject$ to a $\mathcal{S}ubject$. The $rule$ triplet is defined as follows:

$$\forall rule \in \mathcal{R}_{app},\ rule = (s,\ r,\ o) \qquad (1)$$

where $s = \mathcal{S}ubject,\ r \in \mathcal{R}ight,\ o \in \mathcal{O}bjects$

We define a mobile application as a $\mathcal{S}ubject$. Each mobile application is associated with the unique identification number can be used as a $\mathcal{S}ubject$.

$$\mathcal{S}ubject = Mobile\,Application\,ID \tag{2}$$

$\mathcal{O}bjects$ are the user-related data, such as e-mail, contact list, name and surname, phone number, address, social networks friend list, etc. As the permission system is data-centred, the definition of data should be as precise as possible.

$$\mathcal{O}bjects = \{Phone\#,\ Name,\ Contacts,\ \cdots\} \tag{3}$$

Each application needs to use a piece of personal data to perform a particular action and with a specific goal. Users give the $\mathcal{R}ight$ to the application according to this action and this goal. To define $\mathcal{R}ight$ we have to introduce $\mathcal{A}ction$ and $\mathcal{P}urpose$.

Each action is one of all the actions, denoted $\mathcal{A}ctions$, that can be carried out on user private data by the application: load, read, modify, store and transfer. We define the $\mathcal{A}ctions$ as follows:

$$\mathcal{A}ctions = \{Read, Modify, Load, Store, Transfer\} \tag{4}$$

where

$\mathcal{R}ead$ is a read-only access to the data that is already stored on the phone.

$\mathcal{M}odify$ is an action permitting the replacement or update of a piece of personal data already stored in the system.

$\mathcal{L}oad$ represents an action bringing new information to the phone from a distinct server, Internet or mobile sensor such as GPS, etc.

$\mathcal{S}tore$ action indicates a new piece of private data will be saved on the device.

$\mathcal{T}ransfer$ action indicates some private data is transmitted from the device to the server or another device.

$\mathcal{P}urpose$ is assigned by the application developer and depends on the service. For example, purpose could be 'retrieve forgotten password', 'display on the screen', 'calculate the score', 'send news', 'retrieve nearest restaurant' and 'attach to the message'.

$$\mathcal{P}urpose = \{Retrieve\,forgotten\,password,\ \cdots\} \tag{5}$$

We define a permission right, denoted $\mathcal{R}ight$, for all actions except the $Store$ action as a combination of one element of $\mathcal{A}ctions$ and one element of $\mathcal{P}urposes$.

To respect the minimisation principle, any personal data should be stored on a mobile device only the period of time that is necessary for the functionality. We define the set of rights, denoted $\mathcal{R}ight$, with the action equal to $Store$ having an additional parameter, $time$, informing about the time storage. We define the period $[0, T]$ as an application lifetime.

$$\forall r \in \mathcal{R}ight,\ r = \begin{cases} (action,\ purpose) & if\,(r*) \\ (action,\ purpose,\ time) & if\,(r**) \end{cases} \tag{6}$$

(r*) $action \in \mathcal{A}ctions - \{Store\}$

(r**) $action = Store$

where $purpose \in \mathcal{P}urposes,\ time \in [0, T]$.

The time storage can indicate the number of days, hours or months data is stored or the time regarding the application lifecycle: until the application is closed, until the application is stopped, until the application is uninstalled. All personal data available during the deinstallation of the application is deleted regardless of the defined period, as it cannot exceed the application lifetime.

B. Object: private data

Using existing mobile permissions and mobile forensic techniques on iOS and Android phones we identified some private data that can be accessed on the smartphone by an application. Below is an exhaustive list of personal information that can be found on a mobile phone.

Contact list

1) Type (phone, Facebook, Twitter, Skype) or associated service names
2) Name
3) Surname
4) Nickname
5) E-mail
6) Picture
7) Address
8) Website
9) Company
10) Birthday
11) Job title
12) Significant other

Calendar

1) Calendar name
2) Appointment subject
3) Appointment location
4) Appointment starting date
5) Appointment ending date
6) Appointment starting time
7) Appointment ending time
8) Appointment status
9) Appointment notes
10) Appointment attendees' full names

SMS

1) SMS type (incoming, outgoing)
2) Time
3) Sender's Name
4) Sender's phone number
5) Text content
6) Multimedia content (small images)

MMS

1) MMS type (incoming, outgoing)
2) Time
3) Sender's Name
4) Sender's phone number
5) Text content

6) Multimedia content (images, photos, video, contact information, etc.)

Call logs

1) Call type (incoming, outgoing, missed)
2) Caller's Name
3) Caller's phone number
4) Voice messages

Location

1) Exact location: latitude and longitude
2) Approximate location: latitude and longitude
3) Address
4) Street
5) City
6) Country

Stored Multimedia

1) Type (image, audio, video)
2) Multimedia content
3) META data (date, geolocation)

Other

1) Sensors multimedia (newly created image, audio, video)
2) Mobile phone usage statistics (last used applications, configurations)
3) Device unique id, SIM id
4) Web browser history
5) User accounts information (login, password, tokens)
6) Documents from external or internal storages
7) Currently displayed screen (screen shots)
8) Push messages
9) Bank account information
10) Biometric information
11) Medical records
12) Social networks and other mobile applications' data

We identified 26 Android permissions giving access to personal data including location, accounts, SMS, MMS, camera, audio and video content, mobile user activities, calls, contacts, calendar, saved documents, external storage data and screen shots. Android group Personal info contains only 16 read and write permissions where only 5 permissions give read access to personal data. Other permissions are distributed between Location, Messages and Hardware control groups. One can see that very few permissions exist on Android to protect personal data, compared to the large amount of personal data that can be available on a mobile device.

C. Permission use restrictions

To reduce the flexibility of permission usage by an application and to give more control to the user, we propose to add to each rule several simple restrictions. Each permission is associated with permission restrictions under which the user accepts the permission. Restrictions should be simple for the user to set up.

Each $\mathcal{R}estriction$ contains an action type from the set of action types denoted $\mathcal{A}ctionTypes$. We define two action types: automatic action, denoted $\mathcal{A}utomatic$, and an action that will be explicitly launched by the user, denoted $\mathcal{U}serEvent$

$$\mathcal{A}ctionTypes = \{Automatic, UserEvent\} \tag{7}$$

$\mathcal{U}serEvent$ action is launched via the user interface by the user triggering a particular event. $\mathcal{A}utomatic$ action is launched by an application and can include regular access, update and transfer of the data or automatic insertion of a complementary piece of data (e.g., automatic attachment of the geolocation data to the message, automatically fill in the form, automatically synchronise the data with the server, etc.). Automatic action can be triggered without any action by the user.

$\mathcal{U}serEvent$ restrictions attach the permission to a particular user event. We define $\mathcal{R}estriction$ for the $\mathcal{U}serEvent$ action type as follows:

$$\begin{aligned} &\forall res \in \mathcal{R}estriction, \\ res = (rule, action-&type, user-trigger-event) \end{aligned} \tag{8}$$

where $action-type = UserEvent$, $rule \in \mathcal{R}_{app}$, $action-type = UserEvent$ and $user-trigger-event$ is a concrete user event intercepted by an application.

$\mathcal{A}utomatic$ action can be triggered by the system with a certain frequency or can be associated with a trigger event (e.g., send message, create new message, etc.) or both. We define $\mathcal{R}estriction$ for the $\mathcal{A}utomatic$ action type as follows:

$$\begin{aligned} &\forall res \in \mathcal{R}estriction, \\ res = (rule, action-&type, frequency, event) \end{aligned} \tag{9}$$

where $action-type = Automatic$, $rule \in \mathcal{R}_{app}$, $frequency$ represent the number of times per day/week/month the action can be performed by an application; $event$ is an event associated to the action launch: application is opened, screen is shown, message is sent (button is clicked), form is filled in, etc.

$Frequency$ is not a mandatory parameter. One permission can have several restrictions: it can be associated with several different user or application events.

D. Permission state

Each $rule$ should be explicitly asked of the user to be assigned. Thus, each $rule$ has a $\mathcal{S}tate$: granted or revoked. The rule is granted with corresponding restrictions, the rule is revoked entirely. To respect the Privacy by Default notion the default $\mathcal{S}tate$ of the permission in installed applications is $Revoked$. Only the user can modify the $\mathcal{S}tate$ of permission with an explicit action via the user interface.

We propose to define the $\mathcal{S}tate$ as follows:

$$\begin{gathered} \mathcal{S}tate(rule, time) = \begin{cases} Granted, & user\, accepts\, the\, rule \\ Revoked, & user\, declines\, the\, rule \end{cases} \\ \mathcal{S}tate(rule, 0) = Revoked \end{gathered} \tag{10}$$

where $rule \in \mathcal{R}_{app}, time \in]0, T]$.

The $State$ of a rule $r1 \in \mathcal{R}_{app}$ changes over the application lifetime. The diagram in Figure 1 shows an example of state modification.

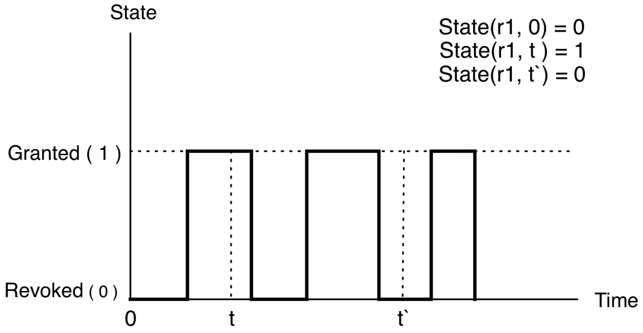


Figure 1. Example of state modification diagram for a given permission

E. User control

User should have a choice of granting the permission permanently, of revoking the permission permanently or of confirming the permission usage with the user each time. We introduce the $rule\ Check$.

$$Check(rule, time) = \begin{cases} True, & confirmation\, required \\ False, & no\, confirmation \end{cases}$$
$$Check(rule, 0) = True \quad (11)$$

where $rule \in \mathcal{R}_{app}$, $time \in \,]0, T]$

If the $Check$ parameter is set to $True$, than the $State$ passes to $Revoked$ and is ignored by the system. The permission is granted or revoked each time by the user via the user interface allowing execution of the functionality, thereby the $Restriction$ of the permissions is also ignored.

If the $Check$ parameter is set to $False$, the system verifies the permission $State$ and $Restriction$ in order to execute the functionality.

To respect the 'Privacy by Default' notion we set the default $Check$ parameter to $True$.

F. Permissions interconnection

Each permission is associated with the purpose, thereby each permission is associated with one particular functionality. Several permissions may be needed to assure one functionality and the developer can give the user a choice of using one permission or another. Several permissions can be grouped by functionality in two ways: all permissions are needed to assure the functionality.

We define the $GroupType$ parameter as follows:

$$GroupType = \{ALL, ONE\} \quad (12)$$

where ALL shows all permissions are necessary to assure the functionality. ONE shows only one of the listed permissions is necessary to achieve the functionality.

The ALL parameter can be expressed as a single permission that should be accepted or declined by the user or by one activation button grouping all permissions. To respect the minimisation principle, all permissions linked to a particular functionality should be Revoked if at least one permission was declined by the final user.

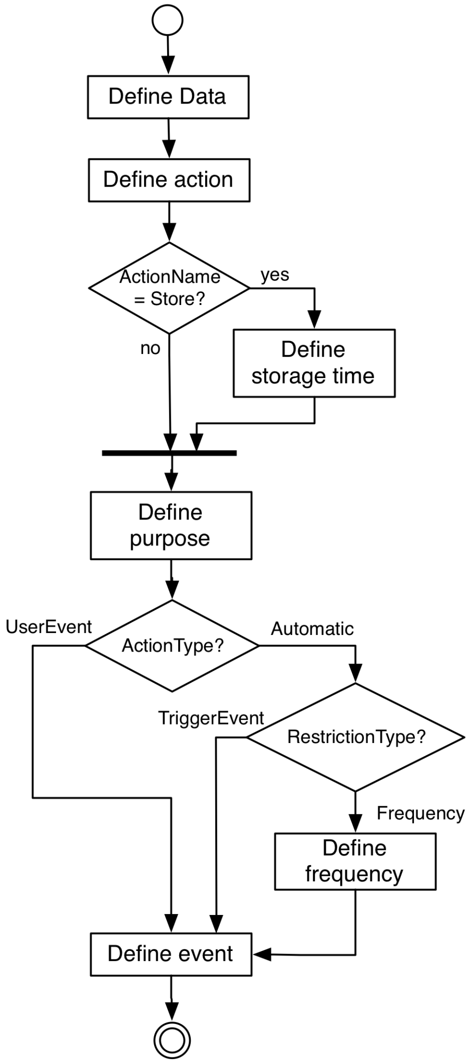


Figure 2. Activity diagram for the rule definition

The ONE parameter can be expressed by the user interface as a group of radio buttons or as one permission with a drop-down list(s) on parameters that differ from permission to permission (e.g., data, usage frequency). If at least one permission is given by the user, the corresponding functionality will be assured and other permissions will be Revoked. For example, an application can assure the service with different types of geolocation data: latitude and longitude, city and the street name and the city only. Developers can propose that the user choose one of the types of geolocation data.

Several permissions can be grouped to add dependencies and an acceptance rule. We define the $Group$ parameter as follows:

$$\forall group \in \mathcal{G}roup$$
$$group = (group - type, \{rule_x, rule_y, \cdots\}) \quad (13)$$

where $group - type \in \mathcal{G}roupType$; $rule_x \in \mathcal{R}_{app}$; $rule_y \in \mathcal{R}_{app}$

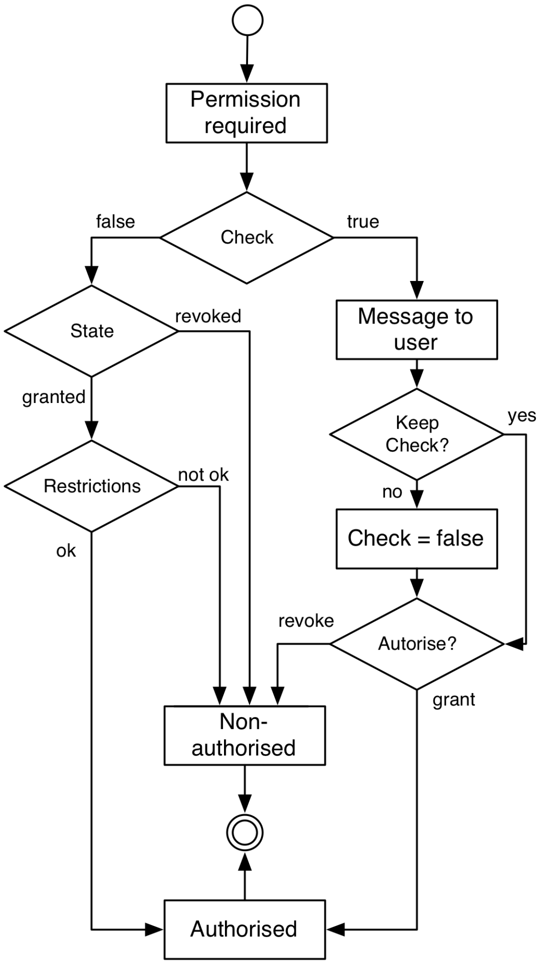


Figure 3. Activity diagram: permission management.

G. Permission in action

The developer should define the permission for all personal data ($\mathcal{O}bject$) used in the application ($\mathcal{S}ubject$) before making the application available on the market. Permission restrictions and permission groupes should also be defined. Figure 2 shows the recapitulative schema of the permission definition.

The permission ($rule$) is stored inside the application with its current $State$ and $Check$ parameters. The default $State$ is $Revoked$. The default $Check$ is $True$. Developers should verify that the permission is fully $displayed$ with the corresponding object, action, purpose and restrictions and $requested$ at least once and that the user is able to grant or revoke this permission. Finally, the user should stipulate the settings with all $rule \in \mathcal{R}_{app}$ to be able to $Grant$ or to $Revoke$ individual permissions in later use.

The simple privacy policy can be generated from the list of defined rules and added to the store.

The activity diagram in Figure 3 shows the permission management cycle from the permission request to the permission usage authorisation/non-authorisation.

When one permission is required by the application, systems first verifies the parameter $Check$. If $Check$ is set to $True$ the system generates the message including all information about the required permission. Users can accept or to decline the permission as well as switch permission management to automatic (Set $Check$ to $False$). According to the user answer the permission is authorised or non-authorised. $Check$ is set to $False$ means an automatic permission management is enabled. System verifies the permission's $Status$. If permission $Status$ is $Revoked$, then the permission is non-authorised. If permission $Status$ is $Granted$, then the systems check corresponding $Restrictions$. If $Restrictions$ are respected, the permission is authorised, otherwise the permission is non-authorised.

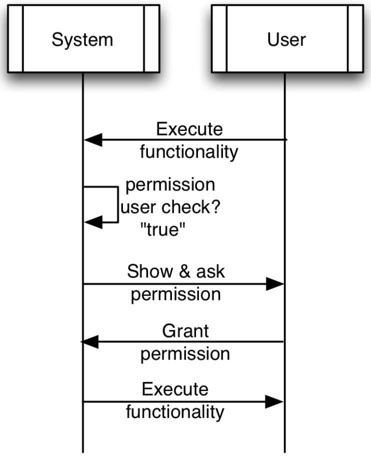


Figure 4. Sequence diagram: first use of one permission or a use or permission in 'user check' mode.

The sequence diagram in Figure 4 shows the one case of permission management when the permission is used for the first time or the $Check$ parameter is set to $True$. For example, the user wants to invite a friend to play a game together, the application needs permission to access the list of contacts and full name with emails to send an invitation mail. System verifies the parameter $Check$ that is set to 'true'. System generates the message for the user explaining the permission needed, the user accepts the permission. Now user can choose the friend he wants to invite.

The sequence diagram in Figure 5 shows the patterns in

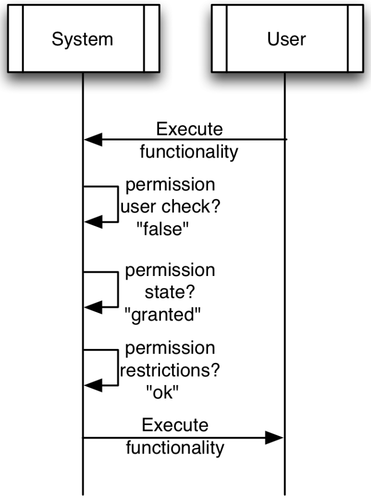

System
User
Execute functionality
permission user check? "false"
permission state? "granted"
permission restrictions? "ok"
Execute functionality

Figure 5. Sequence diagram: use of one permission without user confirmation.

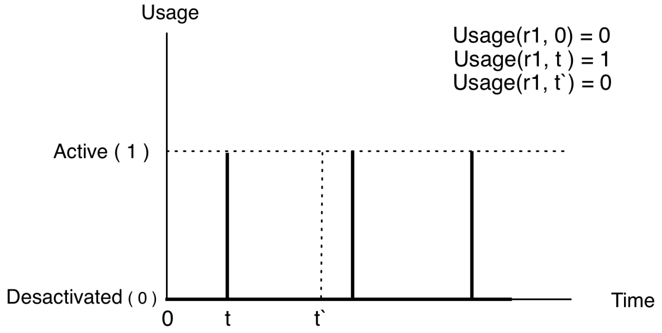


Figure 6. Example of usage modification diagram for a given permission

action when the permission is used automatically without user confirmation: $Check$ parameter is set to $False$. Permission that does not require user confirmation can be used by an application if, and only, the permission status is $\mathcal{G}ranted$ and all $\mathcal{R}estrictions$ are respected. For example, the user wants to automatically attach his geolocation (city) to the first message it sends per day. The permission to load the city from the GPS is needed to attach it to the message. Permission is restricted with a particular event - create new message - and a frequency, once per day. When the user creates a new message, the system verifies the $Status$ of the described permission. As $Status$ is granted, the system verifies the $Restriction$ - permission was not used yet today, therefore, the permission use is authorised.

One can see that the time in which permission can be used by an application is shorter than the time in which permission is Granted. $UserEvent$ action-type permissions can only be used following a particular user event, therefore, permission becomes active only punctually. $Automatic$ permissions are limited by the defined frequency or an event. The usage diagram is depicted in Figure 6

IV. Application

In this section, we propose an example of permission system made for the application of trust evaluation of friends on social networks named Socializer 1.0 [31]. We choose this application because its service is based on private information and cannot be anonymous, the PbD notion should be integrated into this application.

This application needs user friend lists of different social networks (Facebook, Twitter, LinkedIn) and the contact list to view friends and mutual friends to calculate the overlap of friends in different social networks and contact list and to evaluate the trust of Facebook friends.

The following private data can be used by the application:

1) List of contacts from mobile address book: name, surname
2) Facebook friends list: name, surname
3) List of mutual Facebook friends for each Facebook friend: name, surname
4) Twitter friends list: Name, Surname
5) Daily Facebook messages for each Facebook friend
6) Calculated trust score

Contact list is found on the smartphone, therefore, the application needs an $Access$ right.

$$r_1 = (s,\, (Read, p_1),\, ContactList)$$

where s is a $Subject$ defined as the application Socializer 1.0; p_{r1} = calculate the trust scores.

The application will load the user contact list on user event: onClick on the button ”load contact list”.

$$res_1 = (r_1, UserEvent,\, e_1)$$

where e_1 is a user event: onClick on the button ’load contact list’;

Social networking friends lists should usually be retrieved from the server of a given social network, therefore, the load and store actions should be defined. The Facebook friends list with the contact list are essential to assure the overlap and trust functionality.

$$r_2 = (s,\, (Load, p_1),\, FacebookFriendList)$$
$$r_3 = (s,\, (Store, p_1, t_1),\, FacebookFriendList)$$
$$res_2 = (r_2, UserEvent,\, e_2)$$
$$res_3 = (r_3, UserEvent,\, e_2)$$

where t_1 is a storage time defined as: while the application is installed; e_2 is a user event: onClick on the button ”load Facebook friends list”;

For each Facebook friend, the list of mutual friends with the user is necessary for trust calculation.

$$r_4 = (s,\, p_1),\\ FacebookMutualFriendLists)$$
$$res_4 = (r_4, UserEvent,\, e_3)$$

where e_3 is a user event: onItemClick on the user name in the list of users for trust calculation;

A list of friends from other social networks improves the scores of overlap and trust.

$$r_5 = (s,\, (Load, p_2),\, TwitterFriendList)$$
$$r_6 = (s,\, (Store, p_2, t_1),\, TwitterFriendList)$$
$$res_5 = (r_5, UserEvent,\, e_4)$$
$$res_6 = (r_6, UserEvent,\, e_4)$$

where e_4 is a user event: onClick on the button ”load Twitter friends list”; p_2 = improve the trust score;

$$r_7 = (s,\, (Load, p_2),\, LinkedInFriendList)$$
$$r_8 = (s,\, (Store, p_2, t_1),\, LinkedInFriendList)$$
$$res_7 = (r_7, UserEvent,\, e_5)$$
$$res_8 = (r_8, UserEvent,\, e_5)$$

where e_5 is a user event: onClick on the button ”load LinkedIn friends list”;

The second functionality of the application is to evaluate the behaviour of Facebook and Twitter friends to indicate potentially dangerous contacts. The behaviour evaluation is calculated by analysing the messages published by the given friend over time. The application needs a permission to load messages.

$$r_9 = (s,\, (Load, p_2),\, TwitterFriendMessages)$$

$$res_9 = (r_9, Automatic,\, \emptyset,\, e_6)$$

$$res_9 = (r_9, Automatic,\, f_1,\, e_7)$$

where p_3 is a purpose defined as 'calculate the Twitter friends behavior'; f_1 is an action frequency: ones per day; e_6 is a user event: onSlideDown on the list of messages; e_7 is an application event: onApplicationStarted.

$$r_{10} = (s,\, (Load, p_3),\, NewFacebookFriendMessages)$$

$$res_{10} = (r_{10}, Automatic, \emptyset, e_6)$$

$$res_{10'} = (r_{10}, Automatic,\, f_1\, e_7)$$

where p_3 is a purpose defined as 'calculate the Facebook friends behaviour'.

The third functionality proposes to view today Facebook and Twitter messages on the screen for the user.

$$r_{11} = (s,\, (Store, p_4, t_2),\\ TodayTwitterFriendMessages)$$

$$res_{11} = (r_{11}, Automatic,\, f_1\, e_7)$$

$$res_{11'} = (r_{11}, Automatic, \emptyset, e_6)$$

where p_4 is a purpose defined as 'view today Twitter messages'; t_2 is a storage time defined as: one day.

$$r_{12} = (s,\, (Store, p_5, t_2),\\ TodayFacebookFriendMessages)$$

$$res_{12} = (r_{12}, Automatic,\, f_1,\, e_7)$$

$$res_{12'} = (r_{12}, Automatic, \emptyset, e_6)$$

where p_5 is a purpose defined as 'view today Facebook messages';

The user has the option of sharing the scores by posting new messages on Facebook and Twitter. The user can also contribute to the research by sending the anonymized trust and behaviour statistics to the developer. Those actions should be taken with the user's express consent.

$$r_{13} = (s,\, (Transfer, p_6),\\ FacebookFriendTrustScore)$$

$$res_{13} = (r_{13}, UserEvent,\, e_7)$$

where p_6 is a purpose defined as 'share results on Facebook'; e_7 is a user event: onClick on the button 'share'.

$$r_{14} = (s,\, (Transfer, p_7),\\ FacebookFriendTrustScore)$$

$$res_{14} = (r_{14}, UserEvent,\, e_7)$$

TABLE II
TABLE RECAPITULATING PERMISSIONS NEEDED FOR THE APPLICATION (LAST COLUMN IS A PERMISSION GROUP NUMBER)

Object	Action	Purpose	#
Contacts list	Read	Calculate Trust	1
Facebook friends list	Load; Store		
Facebook mutual friends	Load		
Twitter friends list	Load; Store	Improve Trust	2
LinkedIn friends list			3
Twitter messages	Load	Tw. friends behaviour	4
Facebook messages		Fb. friends behaviour	5
Today Tw. messages	Store; (1 day)	View Tw. messages	6
Today Fb. messages		View Fb. messages	7
Trust score	Transfer	Publish to Twitter	8
		Publish to Facebook	9
Trust and behaviour	Transfer	Contribute to research	10

where p_7 is a purpose defined as 'share results on Twitter'.

$$r_{15} = (s,\, (Transfer, p_8),\, AnonymizedTrust)$$

$$r_{16} = (s,\, (Transfer, p_8),\, AnonymizedBehavior)$$

$$res_{16} = (r_{16}, UserEvent,\, e_8)$$

where p_8 is a purpose defined as 'contribute to the improvement of the methodology'; e_8 is a user event: onClick on the button 'help research'.

The final application has 16 rules required by the application for full functionality.

$$\mathcal{R}_{app} = \{r_1, r_2, r_3, \cdots, r_{15}, r_{16}\}$$

Those rules can be combined into groups. The rules r_1, r_2, r_3 and r_4 have a common purpose, all rules should be accepted to achieve the functionality mentioned in the purpose: 'calculate the trust'.

$$group_1 = (ALL,\, \{r1, r2, r3, r4\})$$

Similarly, r_5 should be grouped with r_6 and r_7 with r_8.

$$group_2 = (ALL,\, \{r5, r6\})$$

$$group_3 = (ALL,\, \{r7, r8\})$$

The rules from r_9 to r_{16} should be accepted one by one to achieve the aforementioned purpose (to achieve the functionality). Finally, we obtain 10 permissions to be added to the application to propose control to the user. Table II recapitulates permissions.

To compare with actual permission systems, (a) iOS requires contact list, Facebook and Twitter access permissions. (b) Android requires 'internet', 'read_contacts' and 'get_accounts' access permissions. Facebook and Twitter connections are managed with APIs that require permissions to be declared on the platform, but the permission management will not be available for users in the mobile application by default. iOS permissions give a certain transparency to the user but Android permissions are vague.

We obtained more fine-grained control of the application and the data including permissions to all necessary personal data, actions carried out on this data and corresponding purposes. The recapitulation table (Table II) clearly shows what data are used for what purpose. This kind of table can be added to the privacy policy to improve transparency. Restrictions should also be added to the table. We did not add restrictions to the table due to the limited space.

V. Conclusion and Future Work

We modelled a permission system for a mobile application including Privacy by Design. This permission system is data-oriented, thus, the final user can easily understand what personal data is involved. We include actions that are missing from current iOS and Android permission systems, such as load and transfer, that improve transparency of the application. We also included simple restrictions to better control data use.

The novelty is to include the purpose of the data use in the permission system. The clear purpose will help users to understand better why the data is used and to judge whether this permission is needed. Purpose in permission also forces developers to apply the minimization principle: a developer cannot use the data if he cannot define the clear purpose of usage. The compulsory purpose definition should help guard against the abusive permission declaration 'in case'. Finally, purpose gives the user more fine-grained control, as the same data can be allowed to be used for one functionality but not for another. It is important for our system to integrate clear purpose and not a vague explanation (e.g., 'measure the frequency of application utilization' instead of 'improve user experience').

PbD states that the user should have control over his data and be Privacy by Default, therefore, permissions used in the application are revoked by default. Users should be clearly informed and asked to grant permission. Moreover, users should keep control of permissions during all the application use time, therefore, the permission setting must be available.

Our permission system helps developers to be compliant with the law; it defines what permissions the developer should add to the application, but in the current state it cannot ensure that all necessary permissions are really added. Our pattern indicates to the developer what should be added to the application to be more transparent, but if he decides to transfer data without asking permission, the pattern allows this (even if it is against European law). The privacy policy generated can give the first indication permitting evaluation if the data use is reasonable and the purpose is clear. Manual verification of an application can show the anomaly in permission system use.

We aim to build a framework for the automatic management of a new permission system to simplify the developers' work. We target the Android system first as it is more crucial due to the more open communication and data sharing, and the vagueness of the current permission system.

The impact of new privacy-respective permission systems on users and developers could be measured by conducting real-life experiments. We aim to measure the impact of integration of the new permission system on design and development time, as well as particular situations and difficulties in applying the pattern. We have an additional hypothesis that the explicative application with high transparency improves user experience and leads to more positive perception of the same application, therefore the use of our permission system benefits the application owner.

References

[1] K. Sokolova, M. Lemercier, and J. B. Boisseau, "Privacy by Design Permission System for Mobile Applications," in PATTERNS 2014, The Sixth International Conferences on Pervasive Patterns and Applications, 2014, pp. 89–95.

[2] TRUSTe. Consumer Mobile Privacy Insights Report. [retrieved: Apr., 2011]

[3] A. Cavoukian, "Privacy by design: The 7 foundational principles," 2009.

[4] E. data protection regulators, "Opinion 02/2013 on apps on smart devices," EU, Tech. Rep., Feb. 2013.

[5] K. D. Harris, "Privacy on the go," California Department of Justice, Jan. 2013, pp. 1–27.

[6] W. Shin, S. Kiyomoto, K. Fukushima, and T. Tanaka, "Towards Formal Analysis of the Permission-Based Security Model for Android," in Wireless and Mobile Communications, 2009. ICWMC '09. Fifth International Conference on. IEEE Computer Society, 2009, pp. 87–92.

[7] K. W. Y. Au, Y. F. Zhou, Z. Huang, P. Gill, and D. Lie, "Short paper: a look at smartphone permission models," in SPSM '11 Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices. ACM, Oct. 2011, pp. 63–67.

[8] R. Stevens, J. Ganz, V. Filkov, P. T. Devanbu, and H. Chen, "Asking for (and about) permissions used by Android apps," MSR, 2013, pp. 31–40.

[9] A. R. Beresford, A. Rice, N. Skehin, and R. Sohan, "MockDroid: trading privacy for application functionality on smartphones," in HotMobile '11: Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, ser. HotMobile '11. ACM, Mar. 2011, pp. 49–54.

[10] P. Hornyack, S. Han, J. Jung, S. Schechter, and D. Wetherall, "These aren't the droids you're looking for: retrofitting android to protect data from imperious applications," in Proceedings of the 18th ACM conference on Computer and communications security, ser. CCS '11. ACM, Oct. 2011, pp. 639–652.

[11] M. Nauman, S. Khan, and X. Zhang, "Apex: extending Android permission model and enforcement with user-defined runtime constraints," in ASIACCS '10: Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. ACM, Apr. 2010, pp. 328–332.

[12] Y. Zhou, X. Zhang, X. Jiang, and V. W. Freeh, "Taming Information-Stealing Smartphone Applications (on Android)," in Trust and Trustworthy Computing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 93–107.

[13] S. Holavanalli et al., "Flow Permissions for Android," in Automated Software Engineering (ASE), 2013 IEEE/ACM 28th International Conference on, 2013, pp. 652–657.

[14] J. Tam, R. W. Reeder, and S. Schechter, "Disclosing the authority applications demand of users as a condition of installation," Microsoft Research, 2010.

[15] S. Egelman, A. P. Felt, and D. Wagner, "Choice Architecture and Smartphone Privacy: There's a Price for That," in Proceedings of the 11th Annual Workshop on the Economics of Information Security (WEIS), 2012.

[16] M. Lane, “Does the android permission system provide adequate information privacy protection for end-users of mobile apps?” in 10th Australian Information Security Management Conference, Dec. 2012, pp. 65–73.

[17] P. G. Kelley et al., “A conundrum of permissions: Installing applications on an android smartphone,” in Proceedings of the 16th International Conference on Financial Cryptography and Data Security, ser. FC’12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 68–79.

[18] A. P. Felt, E. Chin, S. Hanna, D. Song, and D. Wagner, “Android permissions demystified,” in CCS ’11: Proceedings of the 18th ACM conference on Computer and communications security. ACM, Oct. 2011, pp. 627–638.

[19] M. Egele, C. Kruegel, E. Kirda, and G. Vigna, “PiOS: Detecting Privacy Leaks in iOS Applications.” 2011.

[20] C. Gibler, J. Crussell, J. Erickson, and H. Chen, “AndroidLeaks: Automatically Detecting Potential Privacy Leaks in Android Applications on a Large Scale,” in Trust and Trustworthy Computing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 291–307.

[21] T. Vidas, N. Christin, and L. Cranor, “Curbing android permission creep,” in In W2SP, 2011.

[22] Y. Zhang et al., “Vetting undesirable behaviors in android apps with permission use analysis,” in CCS ’13: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. ACM, Nov. 2013, pp. 611–622.

[23] W. Xu, F. Zhang, and S. Zhu, “Permlyzer: Analyzing permission usage in Android applications,” Software Reliability Engineering (ISSRE), 2013 IEEE 24th International Symposium on, 2013, pp. 400–410.

[24] W. Luo, S. Xu, and X. Jiang, “Real-time detection and prevention of android SMS permission abuses,” in SESP ’13: Proceedings of the first international workshop on Security in embedded systems and smartphones. ACM, May 2013, pp. 11–18.

[25] J. Vincent, C. Porquet, M. Borsali, and H. Leboulanger, “Privacy Protection for Smartphones: An Ontology-Based Firewall,” in Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 371–380.

[26] S. Bugiel, S. Heuser, and A.-R. Sadeghi, “mytunes: Semantically linked and user-centric fine-grained privacy control on android,” Center for Advanced Security Research Darmstadt, Tech. Rep. TUD-CS-2012-0226, Nov. 2012.

[27] A. P. Felt, K. Greenwood, and D. Wagner, “The effectiveness of application permissions,” in WebApps’11: Proceedings of the 2nd USENIX conference on Web application development. USENIX Association, Jun. 2011, pp. 7–7.

[28] R. Böhme and S. Köpsell, “Trained to accept?: a field experiment on consent dialogs,” in CHI ’10: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, Apr. 2010, pp. 2403–2406.

[29] E. Smith, “iPhone applications and privacy issues: An analysis of application transmission of iPhone unique device identifiers UDIDs,” October 2010.

[30] “Mobile Apps Study,” Future of Privacy Forum (FPF), pp. 1–16, Jun. 2012.

[31] C. Perez, B. Birregah, and M. Lemercier, “A smartphone-based online social network trust evaluation system,” Social Network Analysis and Mining, vol. 3, no. 4, 2013, pp. 1293–1310.

Improving Automated Cybersecurity by Generalizing Faults and Quantifying Patch Performance

Scott E. Friedman, David J. Musliner, and Jeffrey M. Rye
Smart Information Flow Technologies (SIFT)
Minneapolis, USA
email: {sfriedman,dmusliner,jrye}@sift.net

***Abstract*—We are developing the Fuzzbuster system to automatically identify software vulnerabilities and create adaptations that shield or repair those vulnerabilities before attackers can exploit them. Fuzzbuster's goal is to minimize the time that vulnerabilities exist, while also preserving software functionality as much as possible. This paper presents new adaptive cybersecurity tools that we have integrated into Fuzzbuster, as well as new metrics that Fuzzbuster uses to assess their performance. Fuzzbuster's new tools increase the efficiency of its diagnosis and adaptation operations, they produce more general, accurate adaptations, and they effectively preserve software functionality. We present Fuzzbuster's analysis of 16 fault-injected command-line binaries and six previously known bugs in the Apache web server. We compare Fuzzbuster's results for different adaptation strategies and tool settings, to characterize their benefits.**

Keywords-cyber defense, adaptive security, security metrics, filter generation.

I. Introduction

Cyber-attackers constantly threaten today's computer systems, increasing the number of intrusions every year.Firewalls, anti-virus systems, and patch distribution systems react too slowly to newfound "zero-day" vulnerabilities, allowing intruders to wreak havoc. We are investigating ways to solve this problem by allowing computer systems to automatically identify their own vulnerabilities and adapt their software to shield or repair those vulnerabilities, before attackers can exploit them [1]. Such adaptations must balance the safety of the system against its functionality: the safest behavior might be to simply turn the power off or entirely disable vulnerable applications, but that would render the systems useless. To make a finer-grained balance between security and functionality, adaptations must be:

- General enough to shield the entire vulnerability (i.e., not just blocking an overspecific set of faulting inputs).
- Specific enough to minimize the negative impact on program functionality (e.g., by causing incorrect results on valid inputs).
- Efficiently-generated, to minimize the time during which a vulnerability is present or exposed.

These considerations for adaptive cybersecurity pose several challenges, including: how faults are discovered and diagnosed, with and without direct access to source code or binaries; how adaptations are generated from the diagnoses; how the many possible adaptations are assessed and chosen; and how all of these operations are orchestrated for efficiency.

This paper describes strategies for automatically discovering vulnerabilities, diagnosing them, and adapting programs to shield or repair those vulnerabilities. We have implemented these strategies within the Fuzzbuster integrated system for adaptive cybersecurity [2], which includes metrics [3] and metacontrol [4] for self-adaptive software defense. Fuzzbuster uses a diverse set of custom-built and off-the-shelf fuzz-testing tools and code analysis tools to develop protective self-adaptations. Fuzz-testing tools find software vulnerabilities by exploring millions of semi-random inputs to a program. Fuzzbuster also uses fuzz-testing tools to refine its models of known vulnerabilities, clarifying which types of inputs can trigger a vulnerability. Fuzzbuster's behavior falls into two general classes, as illustrated in Figure 1:

1) *Proactive*: Fuzzbuster discovers novel vulnerabilities in applications using fuzz-testing tools. Fuzzbuster refines its models of the vulnerabilities and then repairs them or shields them before attackers find and exploit them.
2) *Reactive*: Fuzzbuster is notified of a fault in an application (potentially triggered by an adversary). Fuzzbuster subsequently tries to refine the vulnerability and repair or shield it against attackers. Reactive vulnerabilities pose a greater threat to the host, since these may indicate an imminent exploit by an attacker.

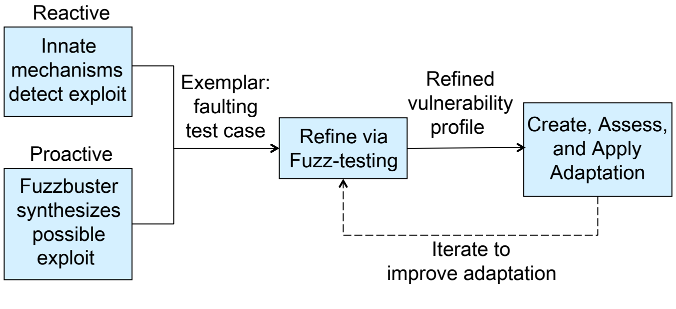

Reactive
Innate mechanisms detect exploit
Proactive
Fuzzbuster synthesizes possible exploit
Exemplar: faulting test case
Refine via Fuzz-testing
Refined vulnerability profile
Create, Assess, and Apply Adaptation
Iterate to improve adaptation

Fig. 1. Fuzzbuster automatically finds vulnerabilities, refines its understanding of their extent, and creates adaptations to shield or repair them.

Fuzzbuster's primary objective is to protect its host by adapting its applications, but this may come at some cost. For example, applying an input filter or a binary patch may create a new vulnerability, re-enable a previously-addressed vulnerability, or otherwise negatively impact an application's usability by changing its expected behavior. This illustrates a tradeoff between functionality and security, and measuring both of these factors is important for making decisions about adaptive cybersecurity.

We begin by outlining related work in Section II and describing FUZZBUSTER's process of discovering, refining, and repairing vulnerabilities in Section III. This supports the adaptation assessment metrics described in Section IV. We then describe FUZZBUSTER's novel diagnosis tools for adaptive cybersecurity in Section V and a novel fault-injection method for generating binaries for testing in Section VI. We evaluate FUZZBUSTER's performance on several experiments in Section VII on real and automatically-injected vulnerabilities in production-grade software. Section VIII summarizes our contributions and future research directions.

II. RELATED WORK

The FUZZBUSTER approach has roots in fuzz-testing, a term first coined in 1988 applied to software security analysis [5]. It refers to invalid, random or unexpected data that is deliberately provided as program input in order to identify defects. Fuzz-testers— and the closely related "fault injectors"— are good at finding buffer overflow, cross-site scripting, denial of service (DoS), SQL injection, and format string bugs. They are generally not highly effective in finding vulnerabilities that do not cause program crashes, e.g., encryption flaws and information disclosure vulnerabilities [6]. Moreover, existing fuzz-testing tools tend to rely significantly on expert user oversight, testing refinement and decision-making in responding to identified vulnerabilities.

FUZZBUSTER is designed both to augment the power of fuzz-testing and to address some of its key limitations. FUZZBUSTER fully automates the process of identifying seeds for fuzz-testing, guides the use of fuzz-testing to develop general vulnerability profiles, and automates the synthesis of defenses for identified vulnerabilities.

To date, several research groups have created specialized self-adaptive systems for protecting software applications. For example, both AWDRAT [7] and PMOP [8] used dynamically-programmed wrappers to compare program activities against hand-generated models, detecting attacks and blocking them or adaptively selecting application methods to avoid damage or compromises.

The CORTEX system [9] used a different approach, placing a dynamically-programmed proxy in front of a replicated database server and using active experimentation based on learned (not hand-coded) models to diagnose new system vulnerabilities and protect against novel attacks.

While these systems demonstrated the feasibility of the self-adaptive, self-regenerative software concept, they are closely tailored to specific applications and specific representations of program behavior. FUZZBUSTER provides a general approach to adaptive immunity that is not limited to a single class of application. FUZZBUSTER does not require detailed system models, but will work from high-level descriptions of component interactions such as APIs or contracts. Furthermore, FUZZBUSTER's proactive use of intelligent, automatic fuzz-testing identifies possible vulnerabilities before they can be exploited.

Other adaptive cybersecurity work focuses on white-box program analysis instead of black-box fuzz-testing. Some of these defenses instrument binaries to change their execution semantics [10] or protect exploitable data [11]. Other approaches adapt binaries to add diversity at compile-time [12] or offline [13], or at load-time [14]. These diversity-based defenses incur comparatively lower overhead and offer statistical guarantees against code-reyse exploits, but unlike FUZZBUSTER, they do not protect the program from the fault itself.

III. BACKGROUND: FUZZBUSTER ADAPTIVE CYBERSECURITY

FUZZBUSTER automatically tests and adapts multiple programs on a host machine by monitoring and adapting the programs' input and output signals. Consequently, programs defended by FUZZBUSTER may be compiled from any high-level programming language or interpreted by a virtual machine, provided the program or virtual machine emits fault signals (e.g., segmentation faults). FUZZBUSTER is designed to accomplish three general goals:

1) *Discovery*: proactively find vulnerabilities within the host's applications.
2) *Refinement*: produce a general, accurate, profile of every vulnerability that FUZZBUSTER encounters.
3) *Adaptation*: provided a refined vulnerability profile, create and assess an adaptation (i.e., patch or filter), and apply it if it improves the state of the application.

FUZZBUSTER is a fully automated system, and it uses a threat-based control strategy [4] to orchestrate its tools (see Table I) in pursuit of these goals.

When FUZZBUSTER discovers a fault in an application— or when it is notified of a *reactive* fault triggered by some other input source— it represents the fault as an *exemplar* that contains information about the system's state when it faulted, as shown in Figure 1. Note that FUZZBUSTER is not responsible for fault detection; we assume that other security and correctness mechanisms detect the fault and notify FUZZBUSTER.

An exemplar includes information for replicating the fault, such as environment variables and data passed as input to the faulting application (e.g., via sockets or `stdin`). Some of this data may be unrelated to the underlying vulnerability. For instance, when FUZZBUSTER encounters a fault during the Apache web server experiment discussed in Section VII, it captures all environment variables (none of which are necessary to replicate the fault), and the entire string of network input that was sent to the application (most of which is not necessary to replicate the fault). FUZZBUSTER uses fuzz-testing tools to incrementally refine the exemplar, trying to characterize the minimal inputs needed to trigger the fault. Since time and processing power is limited, FUZZBUSTER uses a greedy meta-control strategy to orchestrate these tools [4].

TABLE I
FUZZBUSTER'S TASKS. (* = NEW CONTRIBUTION)

Discovery actions replicate and discover vulnerabilities:

- `replicate-fault`: Given an exemplar from the host, replicate the fault under FUZZBUSTER's control.
- `gen-exemplar`: Generate an exemplar that might produce a fault.
- `fuzz-2001`: Generate random binary data and use it as input for `stdin`, file i/o, or command arguments [15], [16].
- `cross-fuzz`: Use Javascript and the DOM to fuzz-test web browsers.
- `wfuzz`: Fuzz-test web servers with templated attacks.
- `retrospective-fault-analysis`: Run faulting test cases through input filters to generate new faulting test cases.*

Refinement actions improve vulnerability profiles:

- `env-var`: Identify environment variables that are necessary for a fault.
- `smallify`: Semi-randomly remove data from the faulting input to find faulting substring(s).
- `div-con`: Binary search for a smaller faulting input.
- `line-relev`: Remove unnecessary lines from multi-line faulting input.
- `find-regex`: Compute a regular expression to capture the faulting input.
- `crest`: Given source code, use concolic search to find constraints on the faulting input [17].
- `replace-all-chars`: Replace characters to generalize buffer overflows.*
- `replace-delimited-chars`: Replace delimited characters to generalize embedded buffer overflows.*
- `replace-individual-chars`: Replace single characters to generalize a faulting input pattern.*
- `insert-chars`: Insert characters to generalize regular expressions.*
- `shorten-regex`: Shorten wildcard patterns to find more accurate buffer overflow thresholds.*

Adaptation actions deploy a shield or repair a vulnerability:

- `create-patch`: Given a vulnerability profile, create a patch to filter input channels and environment variables.
- `verify-patch`: Assess a patch created by `create-patch` to ensure that it outperforms a security baseline.
- `apply-patch`: Apply a verified patch.
- `evolve-patch`: Given source code, use GenProg [18] to evolve a new non-faulting program source and binary.

Refinement is an iterative process, where each task improves the *vulnerability profile* that FUZZBUSTER uses to characterize the vulnerability. The refinement process turns the initial (often over-specific) vulnerability profile into a more accurate and general profile. While refining the Apache web server vulnerabilities, FUZZBUSTER uses an environment variable fuzzer to test and remove unnecessary environment variables for replicating the fault, uses input fuzzers to delimit, test, and remove/replace unnecessary network input, and thereby develops a more accurate vulnerability profile.

FUZZBUSTER has several general adaptation methods, including input filters, environment variable filters, and source-code repair and recompilation. These protect against entire classes of exploits that may be encountered in the future. FUZZBUSTER uses each of these by (1) constructing the adaptation, (2) assessing the adaptation by temporarily applying it for test runs, and (3) applying the adaptation to the production application if it is deemed beneficial. FUZZBUSTER may apply multiple adaptations to an application to repair a single underlying vulnerability. In the case of adapting the Apache web server in Section VII, FUZZBUSTER creates input filters based on its vulnerability profiles: it extracts regular expressions that characterize the pattern of faulting inputs, including necessary character sequences (e.g., "Cookie:"), length-dependent wildcards (e.g., ".{256,}?"), and more. FUZZBUSTER then uses these input filters to identify potentially-faulting inputs and then discard them or rectify them, based on the application under test.

To date, our previous work on FUZZBUSTER has described the overall defense framework and integrated tools [19], [2], extended these tools with concolic testing [20], developed a meta-control strategy for mission- and time-sensitive orchestration of proactive and reactive tools [4], and improved the adaptation metrics [3]. This paper extends our previous publications with (1) new strategies for representing and generalizing vulnerabilities within program inputs, (2) new methods for assessing the safety and functionality of program adaptations, (3) new methods for injecting faults in production-level binaries, and (4) empirical results of FUZZBUSTER adaptating real programs with real vulnerabilities, illustrating the practical benefits of FUZZBUSTER's extensions.

IV. ASSESSING ADAPTATIONS

FUZZBUSTER cannot blindly apply adaptations, since they might have a negative impact on functionality or, even worse, they could create new faults altogether. Thus, FUZZBUSTER uses concrete metrics to assess the impact of candidate adaptations on security and functionality. In this section, we discuss FUZZBUSTER's adaptation metrics, and then we describe and evaluate two of FUZZBUSTER's strategies for assessing adaptations.

A. Metrics for Adaptive Cybersecurity

FUZZBUSTER's adaptation metrics are based on *test cases*: mappings from application inputs (e.g., sockets, `stdin`, command-line arguments, and environment variables) to application outputs (e.g., `stdout` and return code). FUZZBUSTER automatically runs test cases to measure the safety and functionality of the programs it defends. A *faulting test case* terminates with an error code or its execution time exceeds a set timeout parameter, while a *non-faulting test case* terminates gracefully. FUZZBUSTER stores the following sets of test cases for each application under its control:

1) *Non-faulting test cases* are test cases that were supplied with an application for regression testing. FUZZBUSTER tracks which of these have correct behavior (i.e., output and return code), and which have different/incorrect behavior, given some adaptations.
2) *Faulting test cases* include exemplars that caused faults on their first encounter, and other faulting test cases encountered while refining the exemplar. FUZZBUSTER tracks which of these have been fixed by the adaptations

created so far, and which are still faulting. There are two specific types of faulting test cases:

a) *Reactive faulting test cases*: encountered by host notification and subsequent refinement (see Figure 1). These pose more of a threat, since the underlying vulnerability may have been triggered deliberately by an adversary.
b) *Proactive faulting test cases*: encountered by discovery and refinement (see Figure 1). These pose less threat, since they were discovered internally and FUZZBUSTER has no evidence that an adversary is aware of them.

FUZZBUSTER calculates two important metrics from these sets of test cases over time:

1) *Exposure* is computed as the number of unfixed faulting test cases over time. This represents an estimated window of exploitability.
2) *Functionality loss* is computed as the number of incorrect non-faulting test cases over time. This represents the usability that FUZZBUSTER has sacrificed for the sake of security.

Since FUZZBUSTER relies on test cases for measuring exposure and functionality, these measurements are only as complete as the set of faulting and non-faulting test cases, respectively. Before FUZZBUSTER has discovered faults or been notified of faults, there are no faulting test cases for any application. As FUZZBUSTER encounters proactive and reactive faults and refines those faults (e.g., by experimenting with different inputs), it will create additional faulting and non-faulting test cases. FUZZBUSTER accumulates these test cases— as well as any regression tests supplied with the application— and automatically runs them as described below to assess potential adaptations. FUZZBUSTER then applies and removes adaptations to fix the faulting test cases and restore the behavior of non-faulting test cases. These adaptations ultimately protect the host against adversaries.

We note that concolic testing tools (e.g., [21], [22]), including those already integrated with FUZZBUSTER [17], [20], can generate sets of test cases from the program's source code; however, since this work focuses on black-box fuzz-testing, FUZZBUSTER uses black-box tools to generate its test cases.

B. Two Adaptation Policies

Not all of FUZZBUSTER's adaptations improve the status of the analyzed program: some adaptations may sacrifice functionality (i.e., change the behavior of non-faulting test cases) without improving exposure (i.e., fixing faulting test cases), others may cause new faults, and still others may have no measurable effect. Using the metrics described above, FUZZBUSTER can apply different adaptation policies to assess whether an adaptation should be applied to a program.

In our first experiment, we compare two different adaptation policies. We describe these policies next, and then provide empirical results to characterize their effect on exposure and functionality.

1) Strict policy: In strict mode, FUZZBUSTER can never change the behavior of a non-faulting test case when adapting a program. What's more, all faulting test cases that correspond to the present vulnerability must be repaired (i.e., test cases corresponding to another vulnerability may still fault). The strict policy thereby only allows adaptations that are complete repairs and that preserve all known functionality, as determined by the available test cases. Once an adaptation is applied, it is never removed.

2) Relaxed policy: In relaxed mode, FUZZBUSTER may sacrifice functionality to fix faulting test cases. The exact balance can be tuned for different applications, but FUZZBUSTER's default priorities are:

1) Fixing reactive faulting test cases.
2) Fixing proactive faulting test cases.
3) Maintaining the behavior of non-faulting test cases.

This means that FUZZBUSTER will tolerate functionality loss (i.e., by changing the behavior of non-faulting test cases) in order to decrease exposure.

C. Experiment: Comparing Adaptation Policies

We conducted an experiment to compare the strict and relaxed adaptation policies. We provided FUZZBUSTER with a faulty version of `dc`, a Unix calculator program. This version of `dc` causes a segmentation fault when either (1) the modulo (%) operator is executed with at least two numbers on the stack or (2) base conversion is attempted with at least two numbers on the stack. Since many different input sequences will produce the fault, we do not expect a single adaptation to address the entire space of faults.

We also provided FUZZBUSTER 25 non-faulting test cases for `dc` to seed the non-faulting test cases set. These were gathered from examples in the `dc` manual and the Wikipedia `dc` entry, with the modulo and base conversion test cases removed.

Results are shown as functionality/exposure plots in Figure 2 and Figure 3. These plots display the following adaptive cybersecurity metrics, as described above:

- The number of faulting test cases FUZZBUSTER has identified through discovery and refinement (solid light red line).
- The number of those faulting test cases that FUZZBUSTER has fixed (dashed light red line).
- Exposure to vulnerabilities (area between light red lines), which FUZZBUSTER should ideally minimize.
- The number of non-faulting test cases FUZZBUSTER has for the application (solid dark blue line).
- The number of those non-faulting test cases whose return code and output behavior is preserved in the patched version (dashed dark blue line).
- Loss of functionality (area between dark blue lines), which FUZZBUSTER should ideally minimize.
- The patches that have been applied.

Figure 2 shows the results of FUZZBUSTER's strict policy. By definition, the strict policy preserves all functionality of the application, so we do not plot non-faulting test cases in Figure 2.

Adaptations are numbered sequentially, starting with "Patch 1" and increasing with each adaptation created. Following the patch are three numbers: (1) number of faulting test cases *created* by the patch; (2) number of faulting test cases *fixed* by the patch; and (3) number of non-faulting test cases *preserved* by the patch. As shown in Figure 2, under the strict policy, FUZZBUSTER created over 109 adaptations in 45 minutes, but only 11 passed the strict assessment criteria and were subsequently applied.

Figure 3 shows the results of the relaxed policy. This is plotted in the same way as the strict results, except we also include the dark blue non-faulting dataset. FUZZBUSTER accumulates non-faulting test cases over time by using fuzz-tools to generate and run test cases that do not produce faults.

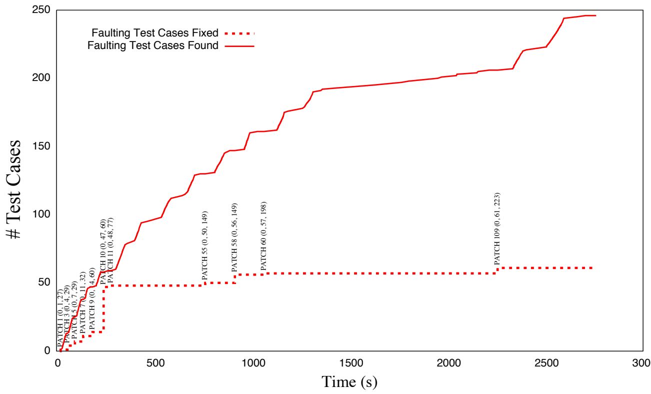


Fig. 2. FUZZBUSTER uses the strict policy to preserve its applications' functionality throughout the course of protective adaptation.

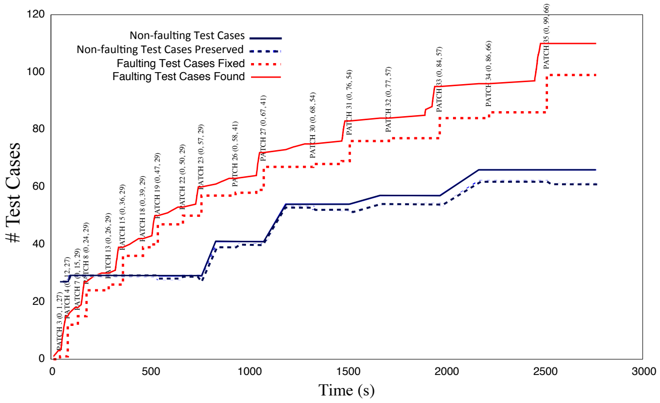


Fig. 3. FUZZBUSTER uses the relaxed policy to sacrifice functionality for the sake of improving security.

As shown in Figure 3, FUZZBUSTER incurs a loss of functionality (gap in between dark blue lines) to reduce its exposure to vulnerabilities. More specifically, it sacrifices up to 10% of its non-faulting test cases to fix faulting test cases, and it restores the behavior of erroneous non-faulting test-cases at multiple points, e.g., around 600s and 950s.

There are several key differences between the strict and relaxed results:

- The exposure gap is much smaller in relaxed mode than in strict mode, indicating more protection over time in relaxed mode.
- In relaxed mode, FUZZBUSTER applied 18 of 35 (51%) created adaptations, compared to 11 of 109 (10%) in strict mode. This means that relaxed mode wasted less time constructing and assessing unused adaptations.
- At the end of each run, relaxed mode yields 110 total faulting test cases, and strict mode yields 245 total faulting test cases. This is because FUZZBUSTER was unable to apply as many adaptations in strict mode, so it discovered more variations of similar faulting test cases (which we verified were due to the same underlying vulnerability).

In relaxed mode, FUZZBUSTER does not fully restore `dc`'s functionality by the end of the 45 minute trial, nor does it restore the functionality after six hours – it retains a 10% loss of functionality. We next describe new fuzz-tools that help close this margin and improve the effectiveness of FUZZBUSTER, as measured by the above adaptive cybersecurity metrics. All of the experiments in the remainder of this paper use the relaxed policy for assessing and applying adaptations.

V. TOOLS FOR DISCOVERY & REFINEMENT

In this section, we describe the new tools we developed to improve FUZZBUSTER's discovery and refinement capabilities. For the sake of comparison, we first review the set of fuzz tools we used in previous work [2], [3], [4].

A. *Previous Fuzz-Tools*

FUZZBUSTER's fuzz-tools included a random string generator for discovering faults (called `Fuzz-2001`) [15], [16] and various minimization (i.e., unnecessary character removal) tools for refining faults.

`Fuzz-2001` quickly constructs a sequence of printable and non-printable characters and feeds it as input to the program under test. This is effective for discovering some buffer overflows, problems with escape characters, and other such problems.

The minimization tools FUZZBUSTER uses to refine vulnerabilities include:

- *smallify*: semi-randomly removes single characters from the input string.
- *line-relev*: semi-randomly removes entire lines from the input string.
- *divide-and-conquer*: Use a binary search to attempt to remove entire portions of the input string.

Each of these tools is designed take a faulting test case as input, and produce smaller faulting test case(s).

Minimization tools can operate in a black-box fashion, where FUZZBUSTER does not have the source code or even access to the binary, since they only require an output signal to determine whether the program faulted.

B. *New Fuzz-Tools*

We now discuss several new tools that we have incorporated into FUZZBUSTER for discovering and refining faults. We then present empirical results comparing the new and existing tools to characterize the effects on the host's exposure to vulnerabilities.

These tools work with *input filter adaptations*; that is, program adaptations that remove content from input data before passing the data to the original program.

1) Retrospective Fault Analysis: We implemented and tested *Retrospective Fault Analysis (RFA)*, a new tool for vulnerability discovery. RFA works by finding the most recent faulting test case such that:

- The test case's input is filtered by the most recent adaptation applied, so some input data has been removed.
- The test case still faults, despite its input being filtered.

RFA then uses the test case— with filtered input— as an exemplar. This effectively allows FUZZBUSTER to fix test cases that still fault, despite incremental adaptations.

To illustrate why this is important, consider the following simplified example, where a program faults if it receives either `CRASH` or `fault` in an incoming message. Some messages may have more than one fault within them, e.g.:

- `Cookie: foo=...CRASH...fault...`
- `Cookie: foo=...faCRASHult...`

This means that FUZZBUSTER can automatically build a filter adaptation to address `CRASH`, but in both of the above cases, there will still be a `fault`. Using RFA, FUZZBUSTER will follow its `CRASH` adaptation with a retrospective investigation of the remaining `fault` test case(s). This produces a more complete analysis of problematic inputs, and it reduces the host's exposure to vulnerabilities, as we demonstrate in the experiments in Section VII.

2) Input Generalization Tools: As described above, minimization tools remove unnecessary characters for a fault. Unfortunately, refining vulnerabilities based on removal alone will tend to produce overspecific adaptations.

Consider the example of IP addresses within a packet header: minimization tools might trim 192.168.0.1 to 2.8.0.1, which might still produce the fault; however, an adaptation based on this model will only be effective when 2, 8, 0, and 1 are all present in the address.

FUZZBUSTER's new generalization tools go the extra step of replacing characters and inserting characters to generalize FUZZBUSTER's regular expression model of the faulting input pattern. This means that FUZZBUSTER will be able to substitute the IP address' digits with other digits to develop a more general, accurate adaptation.

We have implemented the following generalization tools:

- `replace-all-chars`: replaces all characters with different characters, reruns the test case, and then generalizes. This helps determine whether the test case is an instance of a buffer overflow. For example:

  `ABCDEFGH ==> .{8,}`
- `replace-delimited-chars`: splits the input into chunks, using common delimiters, removes and replaces delimited chunks, and then generalizes. For example:

  `host: 1.1.1.1\nCookie ==> .{0,}?Cookie`
- `replace-individual-chars`: removes and replaces individual characters, sensitive to character classes (e.g., letters, digits, whitespace, etc.), and generalizes. For example:

  `GCOJR34A59S94H ==> .*C.*R.*A.*S.*H`
- `insert-chars`: inserts characters in-between consecutive concrete characters, to test and relax adjacency constraints. For example:

  `CRASH ==> .*C.*R.*A.*S.*H`
- `shorten-regex`: reduces character counts within wildcard blocks to provide more accurate buffer overflow thresholds. For example:

  `host: .{951,} ==> host: .{256,}`

We conducted experiments on multiple programs to characterize the effect of generalization tools and RFA. We discuss these experiments and results next.

VI. EVOLVING FAULTY BINARIES FOR TESTING

An adaptive cybersecurity evaluation requires a set of programs to adapt and protect. Adapting production-grade software against known Common Vulnerabilities and Exposures (CVEs) is important for demonstrating realism (see Section VII-C); however, these CVEs do not always cover interesting spaces of program inputs for sensitivity analyses. Conversely, hand-injecting faults into the source code of production-grade software allows us to cover interesting input spaces; however, we aim to avoid hand-tailoring our dataset wherever possible.

Given these considerations, we designed and implemented a system to automatically inject faults into existing production-quality binaries using evolutionary programming. Our fault-injection approach takes the following inputs:

- The C source code of the application.
- A set of non-faulting test cases, where each test case is labeled *positive* or *negative*, and there is at least one test case with each label.

Given these inputs, our fault-injector wraps the positive test cases with shell code to return normally (i.e., 0) if there is no error (i.e., `[$? -lt 128 ]`), and then modifies negative test cases to return *abnormally* (i.e., 1) if there is no error. This transformation modifies the test cases to expect an error for any test cases labeled negative.

Our fault-injector then uses the application source code and transformed test cases as input to GenProg [23], an evolutionary program repair tool. In its normal mode of operation, GenProg generates variants of the program and uses the supplied test cases as a fitness function to select variants for the next round of mutation. By transforming the supplied test cases to expect failure from a subset of working test cases, we effectively make GenProg inject faults someplace in the

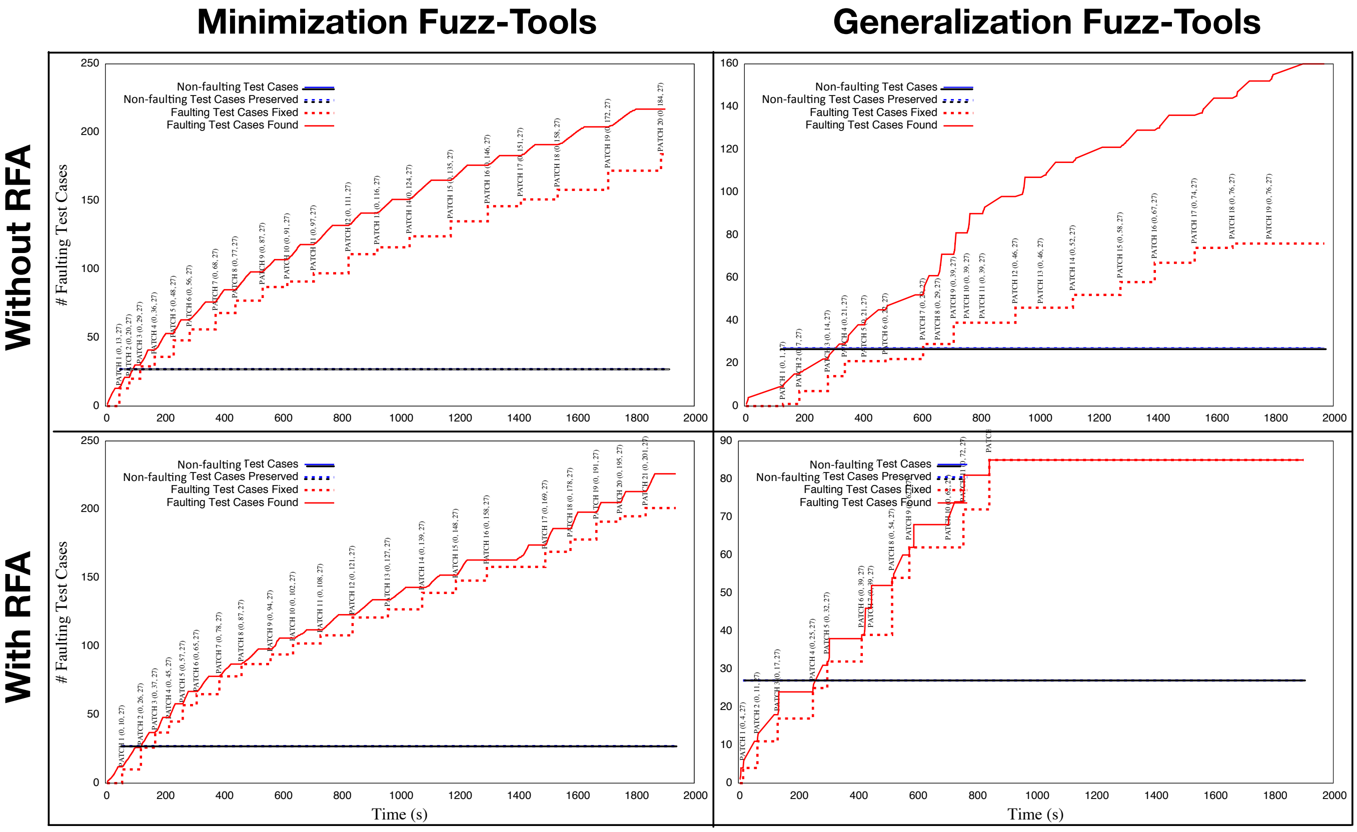


Fig. 4. Results comparing the exposure window of Retrospective Fault Analysis and minimization vs. generalization tools.

code to produce faults for the negatively-labeled cases, and still retain a working program for the positively-labeled cases.

Importantly, evolutionary fault-injection does not produce faults that are limited to the negatively-labeled test cases. Consider the case of the faulty `dc` used in Section IV-C: we supplied a negative test case that used the modulo (%) operation, and a dozen positive test cases that use other arithmetic operations. Evolutionary programming injected a null pointer dereference within an arithmetic helper function used for computing remainders, converting bases, and printing output with a non-decimal radix. Further, for all of these operations to fault, the internal calculator stack had to be non-empty. This demonstrates that evolutionary fault-injection will (1) produce non-trivial faults from simple sets of working test cases and (2) inject faults at arbitrary locations of the program, provided they produce a fault on the given input(s).

We used this automatic fault-injection method to create faulty binaries from the source code of unix command-line applications including `dc`, `fold`, `uniq`, and `wc`. Some of the fault-injected application variants faulted on the vast majority of possible inputs, and gracefully-terminating inputs were very rare. We discarded these variants since they lacked a stable, non-faulting operating mode.

We describe FUZZBUSTER's results adapting 16 fault-injected applications in Section VII-D.

VII. FUZZ-TOOL EXPERIMENTS

We conducted an empirical evaluation on different programs to measure the effect of RFA and the new generalization fuzz-tools. We divide this into four discussions: (1) a comparative analysis of minimization, generalization, and RFA on a single program; (2) an example of FUZZBUSTER sacrificing functionality in order to increase security; (3) a quantitative comparison of minimization and generalization using FUZZBUSTER to shield a web server against known vulnerabilities; and (4) adaptation statistics across multiple programs using FUZZBUSTER with generalization and RFA.

A. Comparative Analysis: Generalization, Minimization, RFA

For this experiment, we used the same fault-injected version of `dc` as in Section IV-C, with a faulty modulo operation. We ran FUZZBUSTER in five settings: with and without RFA, with either minimization or generalization tools (Figure 4); and then with RFA using *both* minimization and generalization tools (Figure 5).

The comparison plots in Figure 4 illustrate the tradeoffs of generalization and RFA. Minimization tools (Figure 4, left) quickly produce overspecific patches. For instance, **PATCH 16** in Figure 4 upper-left plot filters the pattern `.*9.*5.*%.*`. While this is a legitimate example of the fault, it does not characterize the fault in its entirety. By comparison, the generalization patches are slightly more general: **PATCH 6** in the Figure 4 upper-right plot filters the pattern .*d.*%.* (where d duplicates the value on the stack).

Figure 4 also illustrates the effect of retrospective fault analysis. In the RFA trials, the exposure (the area between the red lines) is significantly reduced. This is because FUZZBUSTER often deploys a filter that addresses some – but not all – problems in a faulting input, and then RFA allows FUZZBUSTER to focus on the remainder of the problematic input. For instance, if a single test case has both a modulo operation and a base conversion, filtering out only one of these operations will not repair the test case.

In the setting with both generalization and RFA, FUZZBUSTER filters against the entire vulnerability within 15 minutes; in the other cases, FUZZBUSTER does not level off for over three hours.

Note that in all settings in Figure 4, FUZZBUSTER did not lose functionality of the underlying application, as measured by the correctness of the non-faulting test cases.

Figure 5 shows the results of FUZZBUSTER with both minimization and generalization enabled. It fixes the entire vulnerability and levels off in 18 minutes. Compared to the same condition with minimization disabled (Figure 4, lower right), enabling minimization made FUZZBUSTER spend some unnecessary time attempting to shorten faulting inputs when its generalization tools can find (or have already found) a more general representation of a faulting input pattern.

Also note that the minimization and generalization condition (Figure 5) destroys the functionality of one of the non-faulting test cases with its **PATCH 5**. The overgeneral **PATCH 5** filters out all occurrences of z (push the stack depth onto the calculator stack), which was indeed a character of a faulting input, but when FUZZBUSTER attempted to build a regular expression over its minimization and generalization results, the expression was overgeneral.

These results suggest that FUZZBUSTER's new input-generalizing tools are a suitable replacement for its input-minimizing tools.

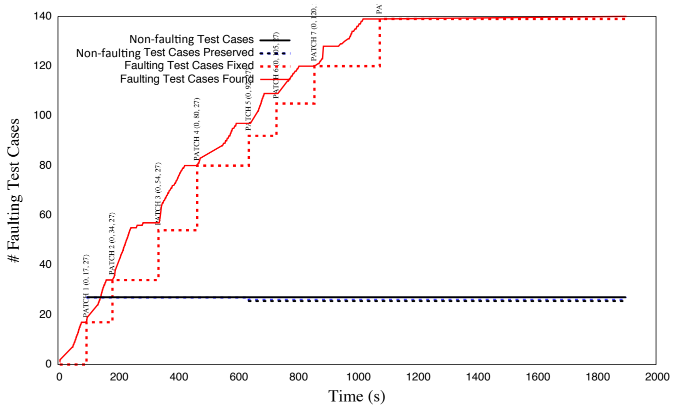


Fig. 5. Results using RFA, minimization, and generalization.

B. Sacrificing Functionality to Increase Security

We ran another FUZZBUSTER trial on a different fault-injected version of the dc binary. This version faulted whenever an arithmetic operation is invoked on an empty stack, so for instance, the sequence ``9 5 +'' would not fault, but the inputs ``+'' or ``4 n +'' would fault due to an empty stack (``n'' pops the stack).

The results are shown in Figure 6. Using generalization tools and RFA, FUZZBUSTER isolates individual arithmetic operations and generates filters for each, ultimately disabling its arithmetic operations to prevent any faults. Note that almost every adaptation has an adverse impact on program functionality, but by design, these are acceptable losses to increase safety of the host.

TABLE II
FUZZBUSTER'S REACTION TIME ON CVES OF THE APACHE WEB SERVER.

CVE	RT (Min.)	RT (Gen.)	Speedup
2011-3192	96	4	24x
2011-3368-1	53	10	5x
2011-3368-2	32	10	3x
2011-3368-3	77	11	7x
2012-0021	36	3	12x
2012-0053	30	7	4x

Reaction times reported in minutes; speedup reported as quotient.

C. Adapting a Web Server

We conducted FUZZBUSTER experiments on known Common Vulnerabilities and Exposures (CVEs) on the Apache web server. This demonstrates FUZZBUSTER working on larger production-quality applications with real vulnerabilities, and it shows the generality of FUZZBUSTER and its fuzz-tools.

For each trial, we initialized FUZZBUSTER with the Apache web server as the only application under test. We then sent a faulting message to the server— as dictated by the corresponding CVE— and FUZZBUSTER detected the reactive fault and began its fuzzing. Table II reports how many minutes FUZZBUSTER took to produce an input filter adaptation (from experiment start to patch time) for the corresponding CVE using only minimization tools (i.e., "Min."), only generalization tools (i.e., "Gen."), and the speedup provided by generalization tools.

In addition to producing more general patches, the generalization tools also yield a significant speedup factor between 3x and 24x, and on average, produce useful adaptations in an order of magnitude less time.

For these CVE trials, RFA was not necessary since FUZZBUSTER fixes all faulting test cases with the first patch it produces.

D. Statistics Across Programs

We now present additional results from using FUZZBUSTER with the generalization tools and retrospective fault analysis on 16 fault-injected binaries created using the process described in Section VI.

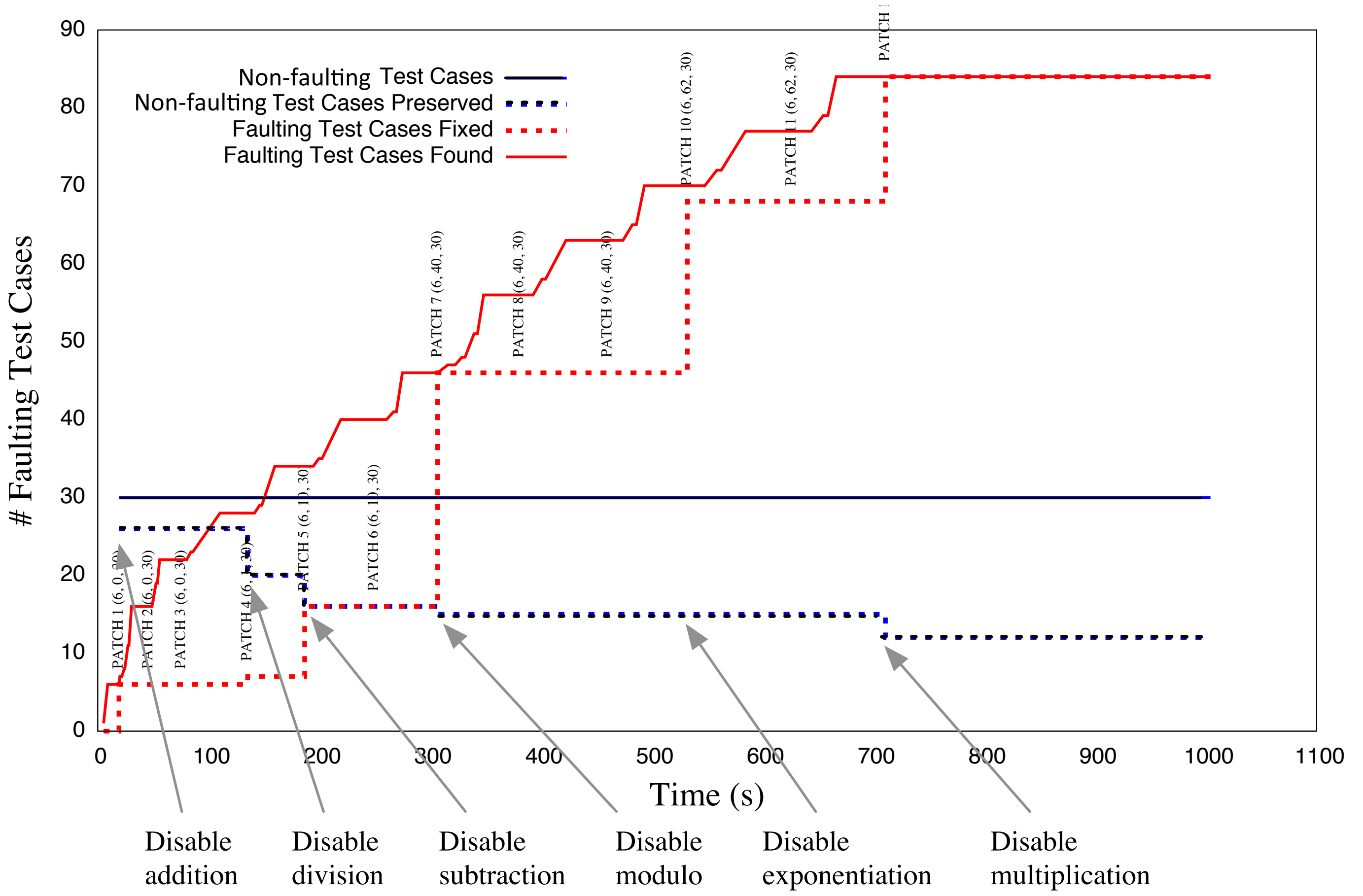


Fig. 6. FUZZBUSTER sacrifices functionality to protect the program against vulnerabilities.

FUZZBUSTER automatically analyzed each faulty binary for two hours, using a mix of proactive fuzz-tools (e.g., Fuzz-2001 and RFA), refinement fuzz tools (e.g., the generalization fuzz-tools), and adaptation strategies (e.g., input filters).

Fuzzing *leveled off* (i.e., FUZZBUSTER patched the entire injected fault, based on our manual analysis of patches) on 10/16 binaries. Of these leveled-off binaries, FUZZBUSTER took an average of 5.87 minutes to level off, and it sacrificed an average of 6% functionality (i.e., by changing the output of non-faulting test cases). FUZZBUSTER retained full functionality on 7 of the 10 leveled-off binaries.

Over all 16 fault-injected binaries, FUZZBUSTER created an average of 8.2 adaptations and applied an average of 7.8, which amounts to a 95% usage of the adaptations it created. Over all binaries, FUZZBUSTER fixed an average of 82% of the faulting test cases and sacrificed an average of 10% functionality during each 2-hour trial. This suggests that when FUZZBUSTER cannot generate a perfect adaptation, it still manages to close the exposure window over time.

VIII. CONCLUSIONS AND FUTURE WORK

FUZZBUSTER is designed to discover vulnerabilities and then quickly refine and adapt its applications to prevent them from being exploited by attackers. This paper uses FUZZBUSTER as a research tool to further the state-of-the-art in measuring and improving adaptive cybersecurity. We presented useful exposure and functionality metrics, we used these metrics to compare and evaluate FUZZBUSTER's self-adaptation policies, and we used the metrics to characterize the benefits FUZZBUSTER's new fuzz-testing tools— retrospective fault analysis and input generalization— aimed at improving the quality and efficiency of adaptive cybersecurity. Further, we described how to automatically inject faults into production-grade software to build datasets for adaptive cybersecurity experimentation.

We presented empirical results of FUZZBUSTER's automated analysis of both fault-injected programs and real CVEs, comparing the vulnerability exposure, functional loss, and reaction time. When analyzing fault-injected programs, the generalization fuzz-tools and RFA reduced vulnerability exposure by a factor of five on fault injected programs, and allowed FUZZBUSTER to shield more of the vulnerability in less time. When analyzing the Apache HTTP server, the new fault generalization tools yielded an order of magnitude speedup in reaction time over the previous fault minimization tools.

Currently, FUZZBUSTER uses a wrapper around the programs it controls, and its wrapper filters all incoming data according to the current adaptations (e.g., input filters) before sending the data to the binary. One next step is to revise the program's binary directly, and embed the input filters as preprocessors.

The generalization fuzz-tools and RFA are all domain-independent strategies, and we demonstrated this by using them to improve program analysis on command-line filter programs (e.g., `wc`), state-dependent standard input programs (e.g., `dc`), and grammar-specific web programs (e.g., Apache HTTP server). The most domain-specific enhancement is the `replace-delimited-chars` tool that uses common delimiters to analyze portions of data. This tool contributed significantly to the speedup of FUZZBUSTER's analysis of HTTP headers in the Apache HTTP server experiment. We believe that we will see additional performance benefits by adding more domain-specific knowledge to FUZZBUSTER, including input grammars (e.g., packet header structure) and deeper application models (e.g., recording application command-line options and values).

Our fault-injector currently requires source code in order to use GenProg [23]. We are investigating fault-injection using evolutionary programming methods that operate directly on assembly code or compiled binaries (e.g., [24]), and do not require source code. This has the advantages of (1) being high-level-language-independent and (2) producing more diverse vulnerabilities at the binary or library level.

We anticipate using the adaptive cybersecurity metrics from this paper to evaluate future design decisions for FUZZBUSTER and other adaptive cybersecurity projects.

ACKNOWLEDGMENTS

This work was supported by The Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory under contract FA8650-10-C-7087. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Approved for public release, distribution unlimited.

REFERENCES

[1] D. J. Musliner, S. E. Friedman, and J. M. Rye, "Automated fault analysis and filter generation for adaptive cybersecurity," in Proceedings of ADAPTIVE 2014: The Sixth International Conference on Adaptive and Self-Adaptive Systems and Applications, June 2014, pp. 56–62.

[2] D. J. Musliner, J. M. Rye, D. Thomsen, D. D. McDonald, and M. H. Burstein, "FUZZBUSTER: A system for self-adaptive immunity from cyber threats," in Eighth International Conference on Autonomic and Autonomous Systems (ICAS-12), March 2012, pp. 118–123.

[3] D. J. Musliner et al., "Self-adaptation metrics for active cybersecurity," in SASO-13: Adaptive Host and Network Security Workshop at the Seventh IEEE International Conference on Self-Adaptive and Self-Organizing Systems, September 2013, pp. 53–58.

[4] D. J. Musliner, S. E. Friedman, J. M. Rye, and T. Marble, "Meta-control for adaptive cybersecurity in FUZZBUSTER," in Proc. IEEE Int'l Conf. on Self-Adaptive and Self-Organizing Systems, September 2013, pp. 219–226.

[5] B. Miller, L. Fredriksen, and B. So, "An empirical study of the reliability of unix utilities," Communications of the ACM, vol. 33, no. 12, December 1990.

[6] C. Anley, J. Heasman, F. Linder, and G. Richarte, The Shellcoder's Handbook: Discovering and Exploiting Security Holes, 2nd Ed. John Wiley & Sons, 2007, ch. The art of fuzzing.

[7] H. Shrobe et al., "AWDRAT: a cognitive middleware system for information survivability," AI Magazine, vol. 28, no. 3, 2007, p. 73.

[8] H. Shrobe, R. Laddaga, B. Balzer et al., "Self-Adaptive systems for information survivability: PMOP and AWDRAT," in Proc. First Int'l Conf. on Self-Adaptive and Self-Organizing Systems, 2007, pp. 332–335.

[9] "Cortex: Mission-aware cognitive self-regeneration technology," Final Report, US Air Force Research Laboratories Contract Number FA8750-04-C-0253, March 2006.

[10] J. Hiser, A. Nguyen-Tuong, M. Co, M. Hall, and J. W. Davidson, "Ilr: Where'd my gadgets go?" in Security and Privacy (SP), 2012 IEEE Symposium on. IEEE, 2012, pp. 571–585.

[11] L. Davi, A.-R. Sadeghi, and M. Winandy, "Ropdefender: A detection tool to defend against return-oriented programming attacks," in Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security. ACM, 2011, pp. 40–51.

[12] M. Franz, "E unibus pluram: massive-scale software diversity as a defense mechanism," in Proceedings of the 2010 workshop on New security paradigms. ACM, 2010, pp. 7–16.

[13] V. Pappas, M. Polychronakis, and A. D. Keromytis, "Smashing the gadgets: Hindering return-oriented programming using in-place code randomization," in Security and Privacy (SP), 2012 IEEE Symposium on. IEEE, 2012, pp. 601–615.

[14] R. Wartell, V. Mohan, K. W. Hamlen, and Z. Lin, "Binary stirring: Self-randomizing instruction addresses of legacy x86 binary code," in Proceedings of the 2012 ACM conference on Computer and communications security. ACM, 2012, pp. 157–168.

[15] B. Miller, L. Fredriksen, and B. So, "An Empirical Study of the Reliability of UNIX Utilities," Communications of the ACM, vol. 33, no. 12, 1990, pp. 32–44.

[16] B. Miller, G. Cooksey, and F. Moore, "An Empirical Study of the Robustness of MacOS Applications Using Random Testing," in Proceedings of the 1st international workshop on Random testing. ACM, 2006, pp. 46–54.

[17] J. Burnim and K. Sen, "Heuristics for scalable dynamic test generation," in Proceedings of the 2008 23rd IEEE/ACM International Conference on Automated Software Engineering, ser. ASE '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 443–446. [Online]. Available: http://dx.doi.org/10.1109/ASE.2008.69

[18] W. Weimer, S. Forrest, C. Le Goues, and T. Nguyen, "Automatic program repair with evolutionary computation," Communications of the ACM, vol. 53, no. 5, May 2010, pp. 109–116.

[19] D. J. Musliner, J. M. Rye, D. Thomsen, D. D. McDonald, and M. H. Burstein, "FUZZBUSTER: Towards adaptive immunity from cyber threats," in 1st Awareness Workshop at the Fifth IEEE International Conference on Self-Adaptive and Self-Organizing Systems, October 2011, pp. 137–140.

[20] D. J. Musliner, J. M. Rye, and T. Marble, "Using concolic testing to refine vulnerability profiles in FUZZBUSTER," in SASO-12: Adaptive Host and Network Security Workshop at the Sixth IEEE International Conference on Self-Adaptive and Self-Organizing Systems, September 2012, pp. 9–14.

[21] P. Godefroid, M. Levin, and D. Molnar, "Automated Whitebox Fuzz Testing," in Proceedings of the Network and Distributed System Security Symposium, 2008, pp. 151–166.

[22] C. Cadar, D. Dunbar, and D. Engler, "KLEE: Unassisted and Automatic Generation of High-coverage Tests for Complex Systems Programs," in Proceedings of the 8th USENIX conference on Operating systems design and implementation. USENIX Association, 2008, pp. 209–224.

[23] W. Weimer, T. Nguyen, C. L. Goues, and S. Forrest, "Automatically finding patches using genetic programming," Software Engineering, International Conference on, 2009, pp. 364–374.

[24] E. Schulte, S. Forrest, and W. Weimer, "Automated program repair through the evolution of assembly code," in Proceedings of the IEEE/ACM international conference on Automated software engineering. ACM, 2010, pp. 313–316.

iaria journals

www.iariajournals.org

International Journal On Advances in Intelligent Systems
issn: 1942-2679

International Journal On Advances in Internet Technology
issn: 1942-2652

International Journal On Advances in Life Sciences
issn: 1942-2660

International Journal On Advances in Networks and Services
issn: 1942-2644

International Journal On Advances in Security
issn: 1942-2636

International Journal On Advances in Software
issn: 1942-2628

International Journal On Advances in Systems and Measurements
issn: 1942-261x

International Journal On Advances in Telecommunications
issn: 1942-2601