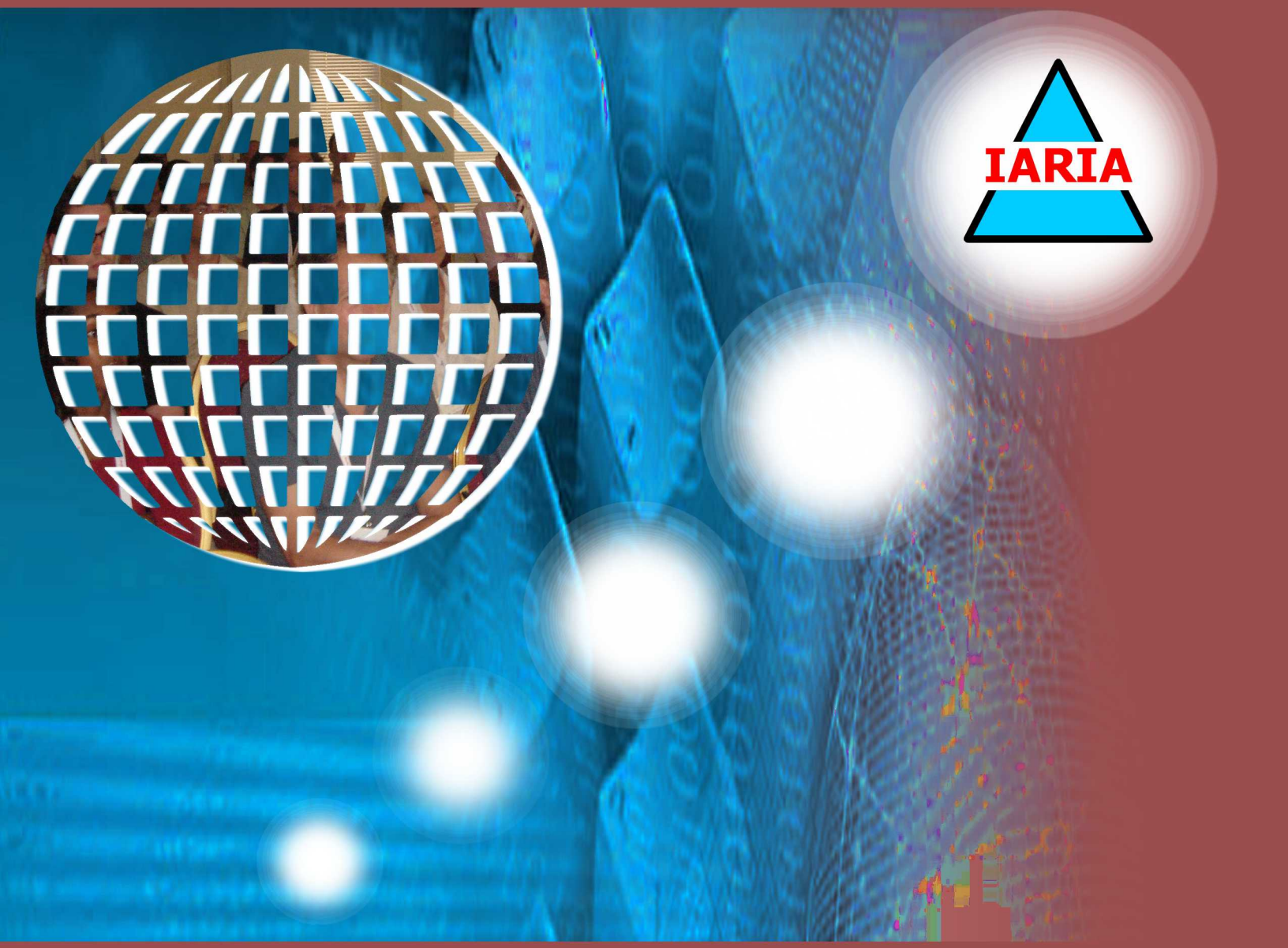
International Journal on Advances in Security



2020 vol. 13 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

Responsibility for the contents rests upon the authors and not upon IARIA, nor on IARIA volunteers, staff, or contractors.

IARIA is the owner of the publication and of editorial aspects. IARIA reserves the right to update the content for quality improvements.

Abstracting is permitted with credit to the source. Libraries are permitted to photocopy or print, providing the reference is mentioned and that the resulting material is made available at no cost.

Reference should mention:

International Journal on Advances in Security, issn 1942-2636
vol. 13, no. 3 & 4, year 2020, http://www.iariajournals.org/security/

The copyright for each included paper belongs to the authors. Republishing of same material, by authors or persons or organizations, is not allowed. Reprint rights can be granted by IARIA or by the authors, and must include proper reference.

Reference to an article in the journal is as follows:

<Author list>, "<Article title>"
International Journal on Advances in Security, issn 1942-2636
vol. 13, no. 3 & 4, year 2020, <start page>:<end page> , http://www.iariajournals.org/security/

IARIA journals are made available for free, proving the appropriate references are made when their content is used.

Sponsored by IARIA
www.iaria.org

Copyright © 2020 IARIA

International Journal on Advances in Security
Volume 13, Number 3 & 4, 2020

Editors-in-Chief

Hans-Joachim Hof,
- Full Professor at Technische Hochschule Ingolstadt, Germany
- Lecturer at Munich University of Applied Sciences
- Group leader MuSe - Munich IT Security Research Group
- Group leader INSicherheit - Ingolstädter Forschungsgruppe angewandte IT-Sicherheit
- Chairman German Chapter of the ACM

Birgit Gersbeck-Schierholz
- Leibniz Universität Hannover, Germany

Editorial Advisory Board

Masahito Hayashi, Nagoya University, Japan
Daniel Harkins , Hewlett Packard Enterprise, USA
Vladimir Stantchev, Institute of Information Systems, SRH University Berlin, Germany
Wolfgang Boehmer, Technische Universitaet Darmstadt, Germany
Manuel Gil Pérez, University of Murcia, Spain
Carla Merkle Westphall, Federal University of Santa Catarina (UFSC), Brazil
Catherine Meadows, Naval Research Laboratory - Washington DC, USA
Mariusz Jakubowski, Microsoft Research, USA
William Dougherty, Secern Consulting - Charlotte, USA
Hans-Joachim Hof, Munich University of Applied Sciences, Germany
Syed Naqvi, Birmingham City University, UK
Rainer Falk, Siemens AG - München, Germany
Steffen Wendzel, Fraunhofer FKIE, Bonn, Germany
Geir M. Køien, University of Agder, Norway
Carlos T. Calafate, Universitat Politècnica de València, Spain

Editorial Board

Gerardo Adesso, University of Nottingham, UK
Ali Ahmed, Monash University, Sunway Campus, Malaysia
Manos Antonakakis, Georgia Institute of Technology / Damballa Inc., USA
Afonso Araujo Neto, Universidade Federal do Rio Grande do Sul, Brazil
Reza Azarderakhsh, The University of Waterloo, Canada
Ilija Basicevic, University of Novi Sad, Serbia
Francisco J. Bellido Outeiriño, University of Cordoba, Spain
Farid E. Ben Amor, University of Southern California / Warner Bros., USA
Jorge Bernal Bernabe, University of Murcia, Spain
Lasse Berntzen, University College of Southeast, Norway
Catalin V. Birjoveanu, "Al.I.Cuza" University of Iasi, Romania
Wolfgang Boehmer, Technische Universitaet Darmstadt, Germany
Alexis Bonnecaze, Université d'Aix-Marseille, France
Carlos T. Calafate, Universitat Politècnica de València, Spain
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Zhixiong Chen, Mercy College, USA

Clelia Colombo Vilarrasa, Autonomous University of Barcelona, Spain
Peter Cruickshank, Edinburgh Napier University Edinburgh, UK
Nora Cuppens, Institut Telecom / Telecom Bretagne, France
Glenn S. Dardick, Longwood University, USA
Vincenzo De Florio, University of Antwerp & IBBT, Belgium
Paul De Hert, Vrije Universiteit Brussels (LSTS) - Tilburg University (TILT), Belgium
Pierre de Leusse, AGH-UST, Poland
William Dougherty, Secern Consulting - Charlotte, USA
Raimund K. Ege, Northern Illinois University, USA
Laila El Aimani, Technicolor, Security & Content Protection Labs., Germany
El-Sayed M. El-Alfy, King Fahd University of Petroleum and Minerals, Saudi Arabia
Rainer Falk, Siemens AG - Corporate Technology, Germany
Shao-Ming Fei, Capital Normal University, Beijing, China
Eduardo B. Fernandez, Florida Atlantic University, USA
Anders Fongen, Norwegian Defense Research Establishment, Norway
Somchart Fugkeaw, Thai Digital ID Co., Ltd., Thailand
Steven Furnell, University of Plymouth, UK
Clemente Galdi, Universita' di Napoli "Federico II", Italy
Birgit Gersbeck-Schierholz, Leibniz Universität Hannover, Germany
Manuel Gil Pérez, University of Murcia, Spain
Karl M. Goeschka, Vienna University of Technology, Austria
Stefanos Gritzalis, University of the Aegean, Greece
Michael Grottke, University of Erlangen-Nuremberg, Germany
Ehud Gudes, Ben-Gurion University - Beer-Sheva, Israel
Indira R. Guzman, Trident University International, USA
Huong Ha, University of Newcastle, Singapore
Petr Hanáček, Brno University of Technology, Czech Republic
Gerhard Hancke, Royal Holloway / University of London, UK
Sami Harari, Institut des Sciences de l'Ingénieur de Toulon et du Var / Université du Sud Toulon Var, France
Daniel Harkins , Hewlett Packard Enterprise, USA
Ragib Hasan, University of Alabama at Birmingham, USA
Masahito Hayashi, Nagoya University, Japan
Michael Hobbs, Deakin University, Australia
Hans-Joachim Hof, Munich University of Applied Sciences, Germany
Neminath Hubballi, Infosys Labs Bangalore, India
Mariusz Jakubowski, Microsoft Research, USA
Ravi Jhawar, Università degli Studi di Milano, Italy
Dan Jiang, Philips Research Asia Shanghai, China
Georgios Kambourakis, University of the Aegean, Greece
Florian Kammueller, Middlesex University - London, UK
Sokratis K. Katsikas, University of Piraeus, Greece
Seah Boon Keong, MIMOS Berhad, Malaysia
Sylvia Kierkegaard, IAITL-International Association of IT Lawyers, Denmark
Hyunsung Kim, Kyungil University, Korea
Geir M. Køien, University of Agder, Norway
Ah-Lian Kor, Leeds Metropolitan University, UK
Evangelos Kranakis, Carleton University - Ottawa, Canada
Lam-for Kwok, City University of Hong Kong, Hong Kong
Jean-Francois Lalande, ENSI de Bourges, France
Gyungho Lee, Korea University, South Korea
Clement Leung, Hong Kong Baptist University, Kowloon, Hong Kong
Diego Liberati, Italian National Research Council, Italy
Giovanni Livraga, Università degli Studi di Milano, Italy

Gui Lu Long, Tsinghua University, China
Jia-Ning Luo, Ming Chuan University, Taiwan
Thomas Margoni, University of Western Ontario, Canada
Rivalino Matias Jr ., Federal University of Uberlandia, Brazil
Manuel Mazzara, UNU-IIST, Macau / Newcastle University, UK
Catherine Meadows, Naval Research Laboratory - Washington DC, USA
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
Yong Man Ro, KAIST (Korea advanced Institute of Science and Technology), Korea
Rodrigo Roman Castro, University of Malaga, Spain
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
Peter Schartner, University of Klagenfurt, Austria
Alireza Shameli Sendi, Ecole Polytechnique de Montreal, Canada
Dimitrios Serpanos, Univ. of Patras and ISI/RC ATHENA, Greece
Pedro Sousa, University of Minho, Portugal
George Spanoudakis, City University London, UK
Vladimir Stantchev, Institute of Information Systems, SRH University Berlin, Germany
Lars Strand, Nofas, Norway
Young-Joo Suh, Pohang University of Science and Technology (POSTECH), Korea
Jani Suomalainen, VTT Technical Research Centre of Finland, Finland
Enrico Thomae, Ruhr-University Bochum, Germany
Tony Thomas, Indian Institute of Information Technology and Management - Kerala, India
Panagiotis Trimintzios, ENISA, EU
Peter Tröger, Hasso Plattner Institute, University of Potsdam, Germany
Simon Tsang, Applied Communication Sciences, USA
Marco Vallini, Politecnico di Torino, Italy
Bruno Vavala, Carnegie Mellon University, USA
Mthulisi Velempini, North-West University, South Africa
Miroslav Velev, Aries Design Automation, USA
Salvador E. Venegas-Andraca, Tecnológico de Monterrey / Texia, SA de CV, Mexico
Szu-Chi Wang, National Cheng Kung University, Tainan City, Taiwan R.O.C.
Steffen Wendzel, Fraunhofer FKIE, Bonn, Germany
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 13, Numbers 3 & 4, 2020

CONTENTS

How to Provide High-Utility Time Series Data in a Privacy-Aware Manner: A VAULT to Manage Time Series Data

Christoph Stach
University of Stuttgart
Institute for Parallel and Distributed Systems
Universitätsstraße 38, 70569 Stuttgart, Germany
e-mail: christoph.stach@ipvs.uni-stuttgart.de

Julia Bräcker
University of Stuttgart
Institute for Biochemistry and Technical Biochemistry
Allmandring 5B, 70569 Stuttgart, Germany
e-mail: julia.braecker@lc.uni-stuttgart.de

Rebecca Eichler, Corinna Giebler
University of Stuttgart
Institute for Parallel and Distributed Systems
Universitätsstraße 38, 70569 Stuttgart, Germany
e-mail: {eichlera, giebleca}@ipvs.uni-stuttgart.de

Clémentine Gritti
University of Canterbury
Computer Science and Software Engineering
Christchurch 8041, New Zealand
e-mail: clementine.gritti@canterbury.ac.nz

***Abstract*—*Smart Services* enrich many aspects of our daily lives, such as in the *Ambient Assisted Living* (*AAL*) domain, where the well-being of patients is automatically monitored, and patients have more autonomy as a result. A key enabler for such services is the *Internet of Things* (*IoT*). Using IoT-enabled devices, large amounts of (partly private) data are continuously captured, which can be then gathered and analyzed by Smart Services. Although these services bring many conveniences, they therefore also pose a serious threat to privacy. In order to provide the highest quality of service, they need access to as many data as possible and even reveal more private information due to in-depth data analyses. To ensure privacy, however, data minimization is required. Users are thus forced to balance between service quality and privacy. Current IoT privacy approaches do not reflect this discrepancy properly. Furthermore, as users are often not experienced in the proper handling of privacy mechanisms, this leads to an overly restrictive behavior. Instead of charging users with privacy control, we introduce *VAULT*, a novel approach towards a privacy-aware management of sensitive data. Since in the IoT *time series data* have a special position, VAULT is particularly tailored to this kind of data. It attempts to achieve the best possible tradeoff between service quality and privacy for each user. To this end, VAULT manages the data and enables a demand-based and privacy-aware provision of the data, by applying appropriate privacy filters which fulfill not only the quality requirements of the Smart Services but also the privacy requirements of users. In doing so, VAULT pursues a *Privacy by Design* approach.**

Keywords*—*time series data; privacy filters; aggregation; interpolation; smoothing; information emphasis; noise; data quality; authentication; permission model; data management.

I. Introduction

This paper extends the work of Stach [1]. In this extended version, we discuss for the first time how data are managed in VAULT and how to determine which privacy filters are appropriate for which Smart Service. In addition, we provide more technical and implementation details on the privacy filters.

The ever-increasing popularity of the *Internet of Things* (*IoT*) is both, a blessing and a curse. On the one hand, sensors built into everyday objects enable to monitor entities (e. g., a machine or a person) permanently and very precisely. Since the gathered data are always tagged with a time stamp, the data of different sources can be combined to obtain a comprehensive chronological profile of the monitored entity. Subsequent analyses can provide even more profound knowledge about the entity [2]. The IoT is therefore an enabler for *Smart Services* from a wide variety of domains, including *Smart Homes* [3], *Smart Cars* [4], and *Smart Health* [5]. Such services are a great benefit for the users as they facilitate their daily life [6].

On the other hand, these great capabilities of such services pose a great danger at the same time. In particular, if the monitored entity is a natural person, his or her privacy is at risk. Users are often not even aware of the coherences between gathered data and derivable insights. However, Smart Services not only have access to the data of a single user but to the data of a vast number of users. This even enables them to learn from the behavior of these users and to predict future behavior patterns of different users [7].

For this reason, the *General Data Protection Regulation* of the EU (*GDPR*, see [8]) tries to provide guidance to meet the interests of both, service providers (in terms of data quality) and users (in terms of privacy requirements) [9]. Nevertheless, the user is faced with the difficult task of balancing service quality and privacy. The more data a user shares with a service, the better is its service quality, as it is thereby able to perform more precise analyses and thus establish a more profound knowledge base. Its users, however, are fully exposed in the process. Whereas, if a user conceals all data that could reveal private information, his or her privacy is protected effectively—yet, the service is practically useless as a result [10].

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Today's privacy approaches for the IoT contribute little to solve this dilemma, as they suffer from three critical flaws:

a) Users are often overwhelmed by these approaches, as the coherences between gathered data and derivable knowledge are not comprehensible. That is, if the user grants a service access to two seemingly harmless data sources, the combination of these two sources might provide new insights [11], [12].

b) These privacy approaches completely ignore service quality. They focus solely on concealing certain, possibly private data, and as a result the service quality is often considerably, yet unnecessarily impaired [13].

c) These privacy approaches are only applicable to certain application scenarios and analysis methods. As a result, users need a variety of different privacy solutions to make all of their Smart Services privacy-aware [14].

To this end, we make the following five contributions:

(1) We introduce a privacy approach towards high-utility time series data, called *VAULT*. VAULT is a concept for the protection of personal data, which achieves a good compromise between service quality and privacy and optimizes both of these aspects. Furthermore, specifying privacy requirements is still very simple for the user.

(2) We present five different privacy techniques that are applied in VAULT. These techniques are tailored to the analysis methods applied to time series data as Smart Services mainly handle such data. Furthermore, we describe how the privacy filters in VAULT which implement these privacy techniques can be realized.

(3) We outline how the data management is realized in VAULT. In addition to privacy-aware data handling, a great focus is also on an efficient data provisioning.

(4) We describe how the quality and privacy requirements are specified in VAULT and present an IoT-compliant way of identifying Smart Services. These are prerequisites for tailored data provisioning, which not only enables an appropriate level of service but also respects the privacy of users.

(5) We describe an implementation of VAULT based on *InfluxDB* [15]. Yet, VAULT is completely independent from its data source, i. e., InfluxDB can be replaced by any data source providing time series data.

The remainder of this paper is structured as follows: In Section II, we introduce a sample use case from the *Ambient Assisted Living* (*AAL*) domain. Using this example, we identify requirements a privacy system has to meet in order to be effective for Smart Services. Section III illustrates how privacy mechanisms operate in principle in IoT environments and why this approach poses a problem for data quality. Then, Section IV discusses selected and representative related work regarding whether they meet the identified requirements. We introduce our concept for VAULT and the applied privacy techniques in Section V. An implementation of this concept is given in Section VI. In Section VII, we assess VAULT according to our identified requirements and carry out a performance analysis. Finally, Section VIII concludes this paper and gives a brief outlook on some future work.

II. Running Example

An application field, in which the IoT facilitates the users' daily routines by having access to highly sensitive data, is the healthcare domain. Sensors enable patients to monitor themselves permanently, while their physicians and other parties involved obtain the processed data tailored to their requirements.

In the health context, there is an IoT application that serves the well-being of the users in every stage of life and every conceivable situation. These applications enable users to achieve a permanent self-measurement [17]. Since these applications often involve gamification aspects, users of all ages are motivated to collect a variety of personal information on an ongoing basis, thereby creating and maintaining a very accurate health profile. This is called the *Quantified Self movement* [18].

However, the possibilities of such applications go far beyond pure self-measurement and a Quantified Self. For instance, the sensors in today's commercially available smartphones are accurate enough to process the recorded data for medical analysis [19]. In addition, a variety of special medical metering devices can be connected to a smartphone, e. g., via Bluetooth. In this way, the applications have access to these health data as well [20].

Although the health data are collected using smartphones, the actual processing of the data often involves an online health platform. Such platforms have three advantages: Firstly, they have almost unlimited resources, so that comprehensive analyses are also feasible. Secondly, data of multiple users are available in such platforms, so that statistical analyses can also be carried out. Finally, these platforms also enable to share data with third parties, for instance with doctors and caregivers [21], [22].

Figure 1 illustrates the technical structure of such an IoT health Smart Service. In accordance with Stach *et al.* [16], the components of such a service can be divided into four layers. More about the technical characteristics of these components can be found in Section III.

It is obvious that such a health Smart Service is highly beneficial for both patients and physicians. Patients are able to carry out necessary medical examinations on their own and only need to see their physicians in emergencies. This significantly reduces the workload of the physicians and allows them to focus on emergency cases [23]. However, there are many other parties that are interested in such data. For instance, insurance companies wish to use these data to tailor insurance premiums more dynamically [24]. In addition, these data could provide scientists (e. g., city planners) with the information they need to create healthier living environments [25].

However, not all of these parties need full access to all health data. Especially since such data are highly sensitive, access

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

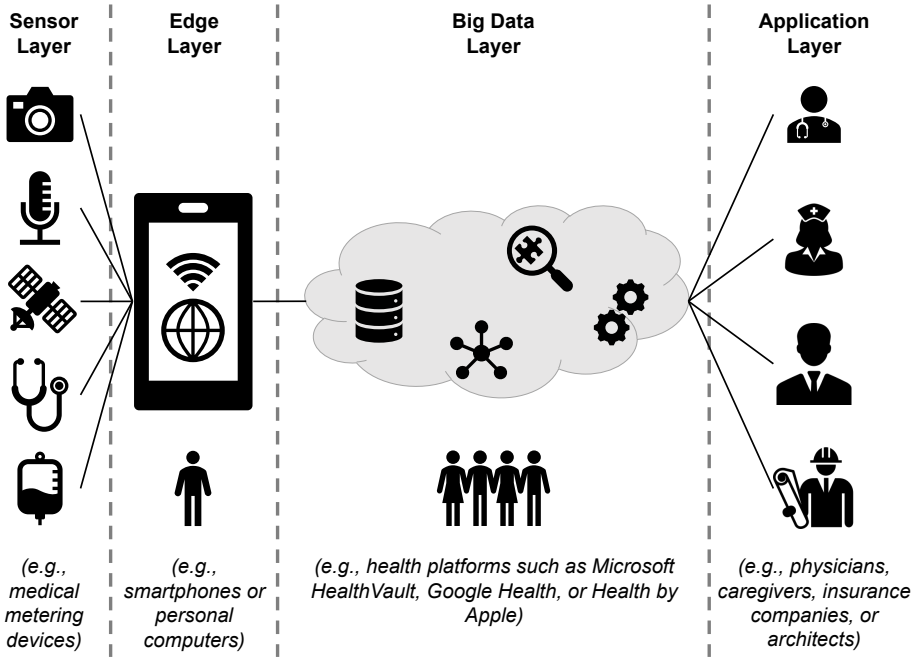


Figure 1. Layered Architecture of an IoT Health Smart Service (cf. [16]).

should be restricted according to the quality requirements of the Smart Services [26].

Application Scenario

In the following, we illustrate this using an AAL use case: Due to an aging population, the *World Health Organization* has introduced the paradigm of *active ageing* to enable elderly people to remain involved in social life [27]. A key aspect in this respect is that they are not pulled from their familiar surroundings (e. g., by accommodating them in a care facility) and that there is no loss of autonomy. AAL achieves this via sensors acting as permanently present but invisible caregivers [28].

An AAL platform offers wide-ranging monitoring services. The health data relevant for such platforms can be effortlessly captured even by technical laymen using conventional sensors. Besides the obvious data acquisition options, such as the use of a GPS sensor, which can be found in every smartphone, for localization, the geomagnetic field sensor can also be used for this purpose as well—that way, even indoor localization is feasible [29]. Additionally, the activity of a user can be determined via a gyroscope and an accelerometer [30].

The consumed bread units (a measurement particularly relevant for diabetics) can be determined with a camera and subsequent image recognition [31]. Even a person's mood can be monitored with a standard microphone based on his or her voice pitch [32], while wearables (e. g., Smart Bands) are able to determine the stress level caused by environmental influences [33]. In addition, special metering devices such as continuous glucose monitoring systems enable a continuous recording and provisioning of blood glucose levels [34]. An example of such a continuous blood glucose monitoring over a period of approximately six months is shown in Figure 2. We use this real-world time series data later to demonstrate the functionality of VAULT.

All these individual measurements can then be combined into a health record object by joining them on their time stamp (see Figure 3). Such a health record can be supplemented with static data, such as annotations to the measurement data or information about contact persons.

Physicians can retrieve these data and are then able to adjust the medication remotely. For some of these health parameters, they require the chronological progression with high accuracy (e. g., blood glucose), while for others an approximate progression is sufficient and single values are negligible (e. g., weight). It is also possible to check remotely, whether the required medication has been taken. Yet, this information is not required to be transferred permanently. It is sufficient to inform physicians if the medicine is not taken several times in a row. Fall detection is realized via wearables. This enables to alert a caregiver immediately if a senior has fallen and needs help. For this purpose, the data from

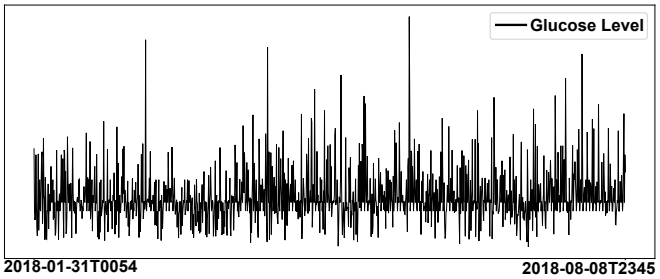


Figure 2. Data from a Continuous Diabetes Monitoring Device.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

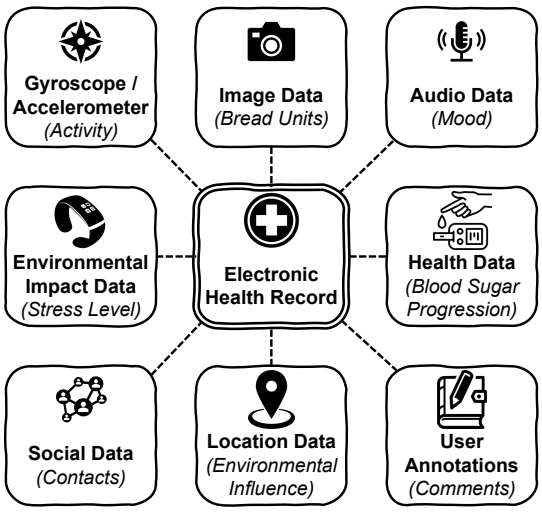


Figure 3. Data Model of an Annotated Electronic Health Record.

the gyroscope, the accelerometer, and the position sensor are analyzed. In addition, the location where the fall occurred has to be determined, e. g., if the "fall" occurred in bed, it may have been a false alarm and the senior just went to sleep. Although location data has to be analyzed for this purpose, the caregiver must not be allowed to access this data. However, relatives with guardianship should be informed of the senior's whereabouts (e. g., if s/he is suffering from dementia and wander around confused and disoriented) [35].

Requirements Specification

This example illustrates that Smart Services gather a variety of private data. The GDPR must thus be observed in such use cases [36]. For instance, it requires *data minimization* [Art. 5(1)(c)]. Caregivers only have to be informed when a senior has fallen, whereas permanent access to his or her location is not required for them. Yet, relatives need access to this data if they are the senior's guardian. This is regulated by the *purpose limitation* [Art. 5(1)(b)]. Service providers have to ensure the *accuracy* of the processed data [Art. 5(1)(d)]. To make this feasible, privacy measures must not arbitrarily manipulate sensor data. Especially when particularly sensitive data, such as health data, is involved, the data subject must give *explicit consent* to their processing [Art. 9(2)(a)]. A solution with respect to these legal obligations is given in Article 25: Technical measures are postulated to ensure privacy compliance, i. e., Smart Services monitor and regulate themselves by default (*Privacy by Design*). To be effective, such a technical privacy solution has to meet the following five requirements:

- **R_1 Individual Privacy Enhancement.** Each user has different privacy requirements. While some people have no concerns about sharing their location data, others consider this kind of data as highly sensitive. Thus, every user has to be able to decide individually what information s/he wants to reveal, i. e., make available to a service.
- **R_2 Utility Preservation.** However, not only privacy requirements need to be considered. Users also have to decide which services they want to use and what data the respective service requires in order to operate. Only if the service receives these data in a sufficient accuracy and quantity, the user receives the expected service quality.
- **R_3 Privacy and Data Quality Harmonization.** Privacy and service quality, however, are by no means independent objectives. Enhancing privacy significantly impairs service quality and vice versa. A privacy system therefore has to consider both aspects equally to achieve *Pareto optimality*.
- **R_4 Privacy Method Adaption.** To make this possible, a privacy system has to be able to adapt its privacy methods to the service quality requested by a user. That is, the privacy system has to select a method which matches a service's specific data quality and quantity requirements.
- **R_5 Dynamic Policy Application.** The application of the privacy requirements has to be dynamic, i. e., before a service gets access to data, its properties must be checked (e. g., a relative only gets access to a senior's location if s/he is his or her guardian at the time of the request).

III. State of the Art

After having identified the requirements towards a technical privacy solution for Smart Services, we now present the four layers of an IoT Health Smart Service (see Figure 1) from a technical point of view. In particular, we aim to specify for each layer, which technical privacy measures can be taken in that respective layer.

Sensor Layer

The sensor layer encompasses all components that can collect data and thus can act as a data source for an IoT Health Smart Service. These are generally very low-level sensors that only serve a specific purpose, e. g., capture blood glucose levels. Their computing power is therefore severely limited and no additional resources such as additional memory or data storages are available to them. As a consequence, no operations that exceed their basic functionality can be executed on these components. This applies especially to third-party applications.

Examples of components that are part of the sensor layer are cameras, microphones, or GPS receivers. However, special medical devices such as continuous blood glucose monitoring systems also belong to this layer.

From a privacy point of view, due to hardware limitations and a lack of capabilities to install privacy protection software on them, users of such devices have no possibilities to control their data unless such a function is explicitly offered by the component. Unfortunately, this is not the case for most of these components. The only privacy control mechanisms on this layer are therefore special privacy-aware connectors that are able to prevent leakage of private data [37], [38].

Nevertheless, the threat level on the Sensor Layer is comparatively low, since on the one hand only a very limited amount of information is captured by each individual component, and on the other hand the data only affects a single data subject, which is typically the owner of the component as well. Moreover, the

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

lack of capabilities to install third-party applications prevents the installation of malware.

Edge Layer

Since the components in the Sensor Layer do not have the required resources, they have to forward the collected data to a device with more computing power for data preprocessing and initial analyses. Thereby it is irrelevant whether they are physically connected to the more powerful device, i. e., whether the sensor is permanently installed in the device, or whether there is a wireless connection. For instance, Bluetooth can be used for that purpose.

Such hub devices are allocated to the Edge Layer. Besides their significantly higher computing power, they are characterized by the fact that they can store larger amounts of data permanently. They also possess connectivity to the Internet and can therefore connect to a health platform (hence the name Edge Layer, as they are on the *edge* to the Cloud).

In principle, any kind of end-consumer product can be considered as such a hub device. This includes smartphones, laptops, or personal computers.

From a privacy point of view, it is both a blessing and a curse that third-party applications can be installed on these devices almost without any restrictions. On the one hand, this enables users to set up privacy control mechanisms that provide a fine-grained permission management in order to ensure that only a bare minimum of data is shared with other applications [39], [40]. On the other hand, however, that is an entry point for malware. Furthermore, their connection to the Internet enables such malware to forward sensitive data to an arbitrary endpoint.

For this reason, the threat level on the Edge Layer is very high. However, this type of device is also characterized by their strong connection to a single user. As a result, data on these devices usually also refer to this single data subject, i. e., the data owner is in control over his or her data as long as they do not leave the Edge Layer.

Big Data Layer

Since the computing power and storage capacity of the devices in the Edge Layer are not sufficient for performing comprehensive long-term analyses, they transfer the collected data to remote servers to this end. These servers are part of the Big Data Layer. In the Big Data Layer Cloud-based health platforms are hosted, such as *Microsoft HealthVault* or *Google Health* [41]. In these platforms a personal health record is maintained for each user which is regularly updated. Besides the simple administration of health data, these platforms also provide continuous data analytics. Since the data of several users are available to these platforms, they can also apply profound data mining, machine learning, and complex event processing techniques to recognize recurring patterns in the data by cross-linking data from different users. That way, further knowledge can be gained.

From a privacy point of view, the threat level is highest for the Big Data Layer. Not only do these Cloud-based health platforms hold a large amount of data from different users, but they also have the capacity to store the data indefinitely. In addition, a user no longer has any physical control over the data once they have been transferred to the platform. Moreover, in a Cloud-based solution, a user does not know where his or her data are processed and stored.

Users have therefore to trust in the reliability of the platform provider. By means of service level agreements and other contract documents, they can protect themselves from a legal perspective. To establish trust in their platforms and prove their fair data usage, platform providers can additionally apply technical privacy control mechanisms [26], [42]. However, these mechanisms have to sustain the data quality so that the functionality of the platform is not impaired.

Application Layer

The insights gained in the Big Data Layer are prepared for presentations tailored to different stakeholders. This includes, for instance, notifications when a certain pattern occurs in real-time data (e. g., a sugar shock is imminent), aggregated reports, or a filtered view on the data. These recipients include physicians, caregivers, or family members of the data subjects, among others.

The devices on which the visualization of the prepared data are rendered belong to the Application Layer. Just like in that Edge Layer, any kind of end-consumer product can be used in the Application Layer, including smartphones, laptops, and personal computers. In contrast to the devices in the Edge Layer, these devices typically are not owned by the data subject.

Therefore, no technical privacy control mechanisms can be applied in this layer since the data subject is not directly connected to these devices. This makes it all the more important that the data subject is able to specify in the Big Data Layer to which third parties the data may be shared with.

Lessons Learned

Table I summarizes the key characteristics of each layer. It considers whether third-party applications can be executed (*Apps*), how much control the user has over the usage of his or her data (*Control*), how many data can be accessed (*Data*), how much computing power is available (*Power*), how many

Table I. Key Characteristics of the Layers of an IoT Health Smart Service (The filling degree of the circles indicates the influence of a certain characteristic on the respective layer.).

	Sensor Layer	Edge Layer	Big Data Layer	Application Layer
Apps	✗	✓	✓	✓
Control	○	◑	◔	○
Data	◔	◑	●	◔
Power	○	◑	●	◔
Storage	○	◑	●	◔
Threat	○	◕	●	◔
User	👤	👤	👥	👥

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

data can be stored persistently (*Storage*), how hazardous the processing can be concerning privacy (*Threat*), and whether the data of a single or multiple users are processed (*User*).

As can easily be concluded from the table, a technical privacy control mechanism for IoT Health Smart Services should be applied in the Big Data Layer.

On the one hand, this is where all the data available to a Smart Service is gathered. If a privacy control mechanism would be applied at an earlier stage, it would be either far too restrictive, e. g., because privacy filters are applied several times, or it would not be comprehensive enough as not all data sources are known at this point. Only in the Big Data Layer all privacy requirements on the part of the users and all quality requirements on the part of the Smart Services are identified.

On the other hand, the Big Data Layer represents the last line of defense before data are passed on to the Application Layer and thus to third parties that are not necessarily known to the data subjects and are therefore completely beyond their control.

IV. Related Work

In the following, we review current privacy approaches for the IoT and assess them with regard to our running example.

Access Control

The most basic approach to ensure privacy is access control. In *role-based access control*, each involved party is assigned to a specific role (e. g., physician). A party can be assigned to several roles at the same time. Access rights to certain data sources are granted to these roles instead of individual users. Although this approach sounds promising at first as there are few roles (compared to the number of parties), and thus the number of access rights which have to be specified is reduced, it is not flexible enough for the IoT due to its fixed predefined roles [43]. Assigning access rights to certain attributes is significantly more dynamic. *Attribute-based access control* validates any kind of attribute at runtime (e. g., attributes that describe the party requesting data access or that party's current context). Data access is only granted if these attributes meet the data subject's authorization requirements [44]. This way, it is possible to model that relatives only have access to a senior's location data if they currently have the guardianship.

Nevertheless, pure access control approaches are far too restrictive and thus severely limit service quality. The user can only make a binary decision—either s/he grants or denies access to a data source. A fine adjustment, however, is not possible (e. g., reduce accuracy of the data or add mock data).

Attribute-based Privacy

To address this problem, a filter can be integrated into a data source. So, particular attributes of the data provided by that source can be filtered out, if they reveal private information. This enables users to specify, e. g., that their medical metering device still provides access to their blood glucose level, but not the blood oxygen level. Each filter can optionally be linked to a *spatiotemporal context* to specify when it should be active [45]. Such a filter can also be tailored to the respective data source. Instead of fully filtering out certain attributes, they can be replaced by mocked but realistic data, in terms of, e. g., value range and distribution [46].

A fundamental problem of these approaches is that they do not take chronological aspects inherent in this kind of data into account. Often, isolated data values do not pose a privacy threat. Only a sequence of single values results in a privacy-relevant pattern (e. g., a sequence of singular gyroscope and acceleration data results in an activity pattern). Yet, users have to filter all data of the concerning attribute in these approaches to ensure that such patterns are concealed. As a result, services depending on this type of data become non-functional.

Pattern-based Privacy

The intent of pattern-based privacy approaches is to conceal complex private information from a Smart Service without unnecessarily restricting its service quality. For this purpose, *Complex Event Processing* (*CEP*) is used. In CEP, no individual sensor values are considered, but higher-order events represented by a sequence of values within a given time window [47]. For instance, the event "senior leaves home" is a sequence of location data representing a motion vector heading away from the house. That way, users specify *private patterns* that must not be revealed and *public patterns* that are critical in terms of service quality. CEP is able to recognize these patterns and then private patterns are concealed by chronologically reordering some of the sensor values. A utility metric identifies the best permutation in terms of maximizing both, privacy and service quality [48].

Pattern-based privacy approaches are therefore particularly effective for maximizing service quality. They can also conceal patterns of any complexity consisting of sequences of individual values. However, such an approach is ineffective with respect to the principle of data minimization. By reordering, all individual values are still sent to the Smart Service. As it is known what kind of information is required by the service (via the public patterns), data could be pre-processed accordingly (e. g., by aggregating or tampering it) without affecting its service quality. For instance, to detect the pattern "senior leaves home", a Boolean statement whether this event occurred is sufficient—the whereabouts prior to this event are not required. Yet, this is not considered by pattern-based privacy approaches.

Statistical Privacy

Differential privacy is applicable to the IoT, e. g., in the context of *Smart Grids* [49]. There, data remains on each user's *Smart Meter*, while energy suppliers only receive aggregated data. It is ensured that no information about an individual user can be derived from the statistical analysis of this data.

Such an approach not only provides a zero-knowledge privacy guarantee for individual users, but also ensures that the accuracy of the data not compromised unnecessarily [50]. Differential privacy can be achieved for both, database systems [51] as well as data stream systems [52].

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

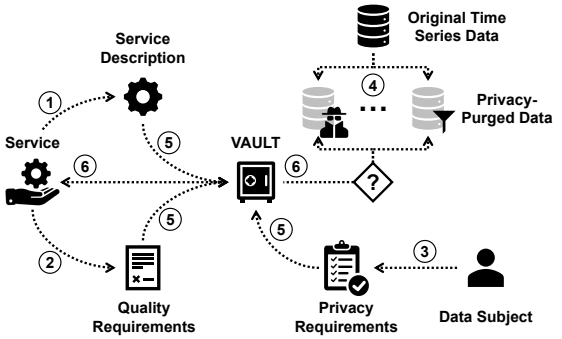


Figure 4. Concept of and Workflow for Data Access via VAULT.

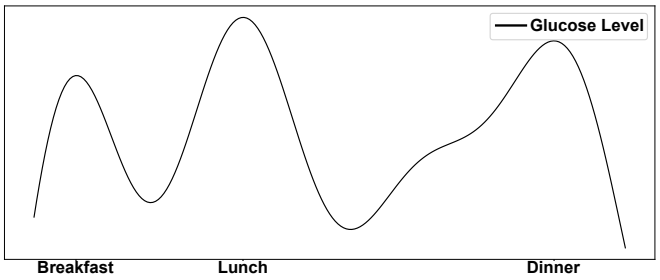


Figure 5. Continuous Diabetes Monitoring Data over the Course of a Day.

Yet, this kind of anonymization is only useful when information about a large group of users is required. It is not applicable to a use case like AAL, as in such a scenario sensor data must be evaluated for each user individually.

V. VAULT Concept

Our review of related work shows that none of these approaches is by itself effective in ensuring both, privacy, and service quality. So, we combine and extend these concepts to provide a privacy concept that is tailored to IoT time series data, called VAULT. According to the findings of Section III, VAULT is positioned in the Big Data Layer.

Figure 4 shows its core concepts and workflow, which are detailed in the following:

Step ① A service description is mandatory that identifies the service, e. g., the service name, its execution environment, or the service owner. This description is used to authenticate to VAULT. Like attribute-based access control, permissions in VAULT are not linked to a specific service, but to a set of its attributes. For instance, different permissions may apply to the same service depending on the country where it is hosted. More information on that authentication and access control can be found in Section VI-A.

Step ② To ensure service quality, a service also has to define its quality requirements. These include, e. g., which data a service requires and with what accuracy these data are required. Thus, the quality requirements correspond to the basic idea of the public pattern.

Step ③ In addition, a data subject specify which permissions are assigned to a certain service. To this end, s/he provides a high-level description of his or her privacy requirements in natural language. Similar to the privacy patterns, s/he only has to describe which knowledge must not be disclosed. A model in VAULT indicates from which data this knowledge can be derived. Based on this model, machine learning can automatically derive permissions from these privacy requirements. More information on that permission management can be found in Section VI-B.

Step ④ As VAULT provides different privacy techniques depending on the respective service (i. e., in accordance with its quality and privacy requirements), the time series data has to be initially prepared accordingly. More information on how the data are managed in this regard is given in Section VI-D.

It has to be mentioned that Step ① to Step ④ are independent tasks and can be carried out in any given order.

Step ⑤ If a service requests data access, VAULT first checks its service description (i. e., attributes of the service) and which permissions (i. e., privacy requirements) are linked to it. They are then consolidated with its quality requirements.

Step ⑥ Based on these two requirement specifications, an appropriate VAULT privacy technique is selected. More information on the privacy filters applied in VAULT can be is given in Section VI-C.

Step ⑦ Subsequently, the request is executed, and the results are sent back to the service.

VAULT relies on existing techniques, which are already used for processing and analyzing time series data, to ensure privacy. As a result, the impact on service quality should be negligible. We discuss the following five such privacy techniques. For this purpose, we use the previously introduced example of continuous diabetes monitoring data. The data set shown in Figure 5 is used to illustrate the respective technique.

Projection, Selection, and Aggregation

The most basic privacy technique used in VAULT is the application of relational algebra operators. A *projection* constrains the number of attributes whereas a *selection* filters out certain tuples of a data source entirely. The impact of these two operators on the result set of a database query is illustrated in Figure 6.

As the data sources we consider in VAULT provide time series data, a selection operator is therefore synonymous with specifying a specific time frame. An *aggregation* can be used to consolidate the analyzed data (e. g., via set operators such as `AVG` or `SUM`). Smart Services use these operators anyway to select the data that is relevant to them and thus reduce the huge amount of available data. VAULT is therefore able to restrict the available data according to the quality requirements of a service via theses operators in order to ensure privacy. For instance, a service gets only access to certain sensor values, certain days, or summarized data.

Listing 1 shows how an SQL query has to be rewritten for this purpose: If a user enters the query shown in Listing 1a, s/he will receive all data stored in the Table “health_record”. A projection ensures in the query shown in Listing 1b that the

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

A	B	C	D
α	β	γ	δ
ε	ζ	η	θ
ι	κ	λ	μ

(a) Raw Data.

A		C	D
α		γ	δ
ε		η	θ
ι		λ	μ

(b) Projection.

A	B	C	D
α	β	γ	δ
ι	κ	λ	μ

(c) Selection.

Figure 6. Impact of a *Projection* and a *Selection* on a Database Query.

user only receives the glucose data stored in the "health_record" Table. The selection in the query shown in Listing 1c causes that only data captured over the course of the last week are returned. Finally, due to the aggregation in the query shown in Listing 1d, only daily average glucose levels are returned.

Data Interpolation

When dealing with sensor data, one has to reckon that sensors occasionally deliver no or incorrect values due to technical problems. To ensure that the data are still processed correctly, strategies must be implemented to deal with these missing and incorrect readings. For this purpose, these incorrect readings have to be substituted with artificial, yet realistic data. On the one hand, *interpolation techniques* can be used to smooth the temporal progression of the values, assuming that the sensor signal describes a continuous function [53]. On the other hand, it is possible to use machine learning to make *predictions* regarding the progression of the values. Missing values or outliers (in terms of values exceeding or falling below a threshold) can then be substituted with these predictions [54].

We use these data cleansing techniques in VAULT to ensure privacy. In certain situations, outliers have a particularly high information value and are therefore considered as particularly sensitive data. Figure 7 shows the time course of a blood glucose level. It can be observed that the level rises particularly high during lunch, which could be a sign that the data subject has eaten dessert. Since this represents an outlier, i. e., an event that occurs only rarely, such a data point holds a particularly high information value. For instance, if a care provider in an AAL program only needs to monitor whether the person is having a meal regularly, the information about the additional dessert can be concealed without causing any problems. To this end, VAULT first uses outlier detection to identify data points with high information value, deletes them, and then fills the resulting gap via *spline interpolation* (red line).

```
SELECT *
FROM "health_record"
```

(a) Query over all Data.

```
SELECT "glucose"
FROM "health_record"
```

(b) Application of a Projection.

```
SELECT *
FROM "health_record"
WHERE time > now()-7d
```

(c) Application of a Selection.

```
SELECT AVG("glucose")
FROM "health_record"
GROUP BY "day"
```

(d) Application of an Aggregation.

Listing 1. Examples of how Queries can be Restricted.

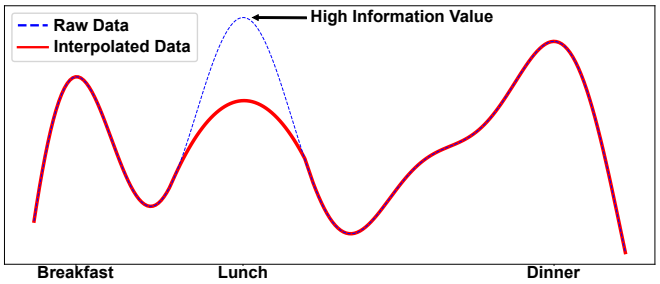


Figure 7. Application of a Spline Interpolation to Time Series Data.

Data Smoothing

While data interpolation is well-suited for eliminating a few isolated outliers, sensor data can also be noisy as a total. Analyzing noisy data is often difficult and leads to poor results. So, the noise component is removed from the data by means of *filters*. Especially if the examined data contains some periodicity, which is often the case with AAL data due to regular daily routines, *Fourier transforms* are well-suited for noise reduction. This creates a band filter effect, i. e., certain interference frequencies can be attenuated [55]. Figure 8 shows the effect of a *Discrete Cosine Transform* on a noisy signal (blue line). The output is a smoothed signal (red line).

However, this data cleansing method can also be used to protect private data. The transform removes details from the time series data and less information is shared with requesting services. Nevertheless, the actual data progression is still available to them with great accuracy. As shown in the figure, smoothing gives a better overview of the blood glucose curve for the six months without revealing any details about particular readings.

Information Emphasis

Using wavelet transform, noise can even be filtered out to such an extent that only data with a high information value remains in the signal (e. g., peaks or turning points). For this purpose, the data progression is compared with a basic function, the so-called *wavelet*. This *window function* defines the weighting of each signal value in subsequent analyses. The *Continuous Wavelet Transform* constantly varies the parameters of this *mother wavelet* to obtain a band of *daughter wavelets*. This facilitates a particularly selective filtering and compression of the data [55].

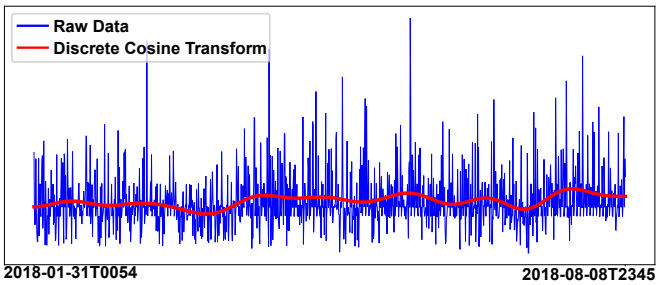


Figure 8. Application of a Fourier Transform to Time Series Data.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

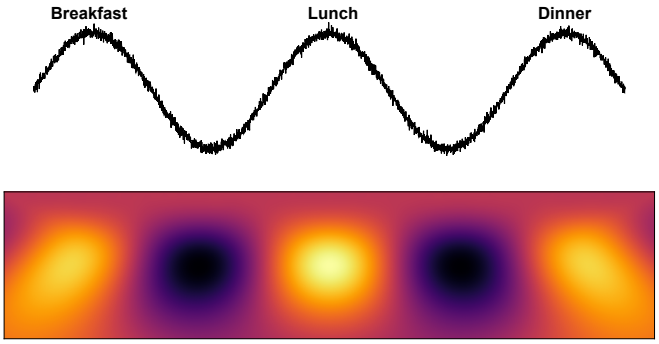


Figure 9. Time-Frequency Representation of Noisy Time Series Data.

In Figure 9, the noisy sensor signal (upper half of the figure) is converted into a *time-frequency representation* (lower half of the figure) using the *Mexican Hat Wavelet* as mother wavelet. Relevant data segments are exposed in this representation (light and dark zones). For instance, if the signal represents blood glucose levels[1], these zones indicate *hypoglycemia* or *hyperglycemia*, respectively. The information about the occurrence of these events is sufficient to generate appropriate recommendations concerning medication and treatment schedule. The exact glucose values need not be disclosed to a caregiver for this purpose. This increases privacy as no details in the data are available to third parties.

Adding Noise

A completely opposite privacy approach is adding noise to a signal on purpose. In Figure 10, *Gaussian noise* is added to formerly noise-free sensor data (blue line). That is, the noise in the resulting data is *Gaussian-distributed* (red line). So, actual values are concealed in a set of corrupted values. Although the general data progression is still noticeable, details and characteristics of the data are hidden by the noise.

For instance, behavior patterns (e. g., "person is having a meal") are thus still recognizable despite the noise, whereas characteristics on what a person has had for lunch are concealed. For instance, food products have a *fingerprint* (i. e., a combination of unique characteristics) that can be used to identify them. One way to identify food products which a person has eaten is by monitoring blood sugar levels [56]. By adding noise to the data, this is no longer possible.

While that initially sounds like a deterioration in data quality, it can even have a positive effect on certain data analyses. For instance, noise can cause *chaotic dynamics* within data. Therefore, if *deterministic chaos* is to be expected in a data set (e. g., data on the course of a disease), but it is not noticeable as too little data are available, adding noise can be useful in this regard to improve analysis results [57].

[1]For the sake of simplicity, the depicted course of blood glucose levels is uniform and regular. However, this is only due to presentation reasons. The assertions and findings presented in this paper also apply to other, irregular courses.

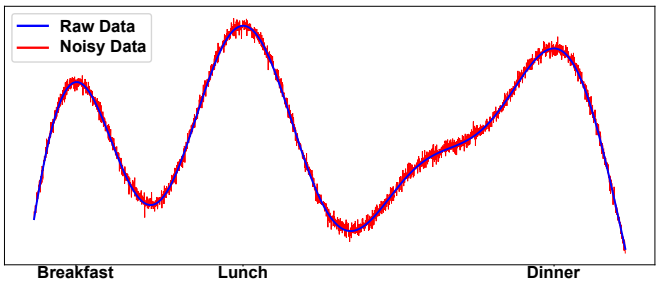


Figure 10. Adding Gaussian Noise to Time Series Data.

VI. VAULT Implementation

In general, there are three implementation strategies for the realization of the VAULT concept, namely *query pre-processing*, *data pre-processing*, and *result post-processing*. Figure 11 shows how these strategies are applied.

Query Pre-Processing: Query pre-processing rewrites queries before execution and adds further constraints to eliminate private information from the result set. This is well-suited for simple privacy techniques such as projection or selection.

By using the Python module *PyPika* [58], queries can be easily rewritten. Listing 2a shows how restrictions, in terms of projections and selections, can be progressively added to a broad incoming query (see Listing 1a). The resulting rewritten query is shown in Listing 2b. Further restrictions, such as aggregations, are also possible with PyPika.

Yet, such query adaptations become complex for more advanced privacy techniques. Then, errors are likely to occur

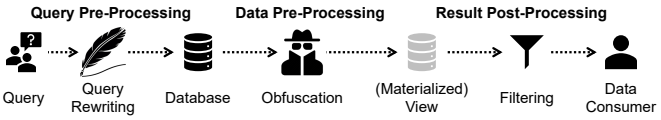


Figure 11. Implementation Strategies for the Privacy Techniques in VAULT.

```python
from pypika import Query, Table, Interval
from pypika import functions as fn

""" initial query over all data """
hr = Table('health_record')
q = Query.from_(hr)

""" adding a projection """
q = q.select('glucose')

""" adding a selection """
q = q.where(hr.time + Interval(days=7) >
    ↪  fn.Now())
query = q.get_sql()
```

(a) Query Rewriting via PyPika.

```
SELECT "glucose"
FROM "health_record"
WHERE "time"+INTERVAL '7 DAY'>NOW()
```

(b) Result of the Query Rewriting.

Listing 2. Exemplary Query Rewriting Process in VAULT.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

when automatically rewriting queries. These errors compromise privacy as well as service quality.

Result Post-Processing: Result post-processing enables a thorough control of a query's result set. That way, it can be filtered before forwarding it to the data consumer.

However, a query can add hidden information to its result set. For instance, if the weight of a person must not be revealed, a data consumer could query all data entries where the weight is x kg (without including the weight itself in the result set). Then, s/he repeats the query and increases x successively. Thus, s/he knows the weight for each entry implicitly, although it never explicitly appeared in the result set. Result post-processing is not able to detect and prevent this.

Data Pre-Processing: Due to the shortcomings of those strategies, we use data pre-processing in VAULT. This strategy pre-processes all data by removing or obscuring private data. Queries are not executed on the original data, but on this purged data. However, this data pre-processing increases the runtime. Yet, as Smart Services often use recurring queries, which are known due to their service descriptions, the runtime can be improved by using materialized views to persist the pre-processed data in advance.

Figure 13 shows how we realized the VAULT concept following the data pre-processing strategy. VAULT introduces a database abstraction layer to strictly isolate services from data sources. From a service's perspective, it therefore seems that it directly interacts with a data source and it is not aware of the privacy techniques applied to the data [59], [60].

Before using a service for the first time, the service provider has to define the quality requirements of the service and the user must specify his or her privacy requirements. As this needs to be done only once (unless requirements change), these steps are not shown in Figure 13.

For this specification, a knowledge model is used. An extract of such a knowledge model is shown in Figure 12. This visualization is based on Stach and Steimle [61].

The knowledge model is based on the principle that data, information, and knowledge are interrelated [62]. Similar to the *DIKW Pyramid* (*data–information–knowledge–wisdom*), our model condenses the raw data of data sources steadily until they become profound knowledge patterns.

Initially, all types of raw data that are available to a Smart Service are specified at the *Data Layer*. As it is irrelevant from which source these data originate, the data sources are not modeled. For instance, an accelerometer can be integrated in both, a Smartphone and a Smart Band. However, since the information content of these two data sources (and thus the potential privacy threat) is identical, we do not differentiate between them in order to keep the knowledge model as concise and comprehensible as possible. Moreover, at this layer, it is not relevant whether a Smart Service actually uses these data—it only matters which data are available. As a result, the knowledge model becomes slightly more extensive than necessary. That way, however, the model remains compatible with future Smart Services that might use these data.

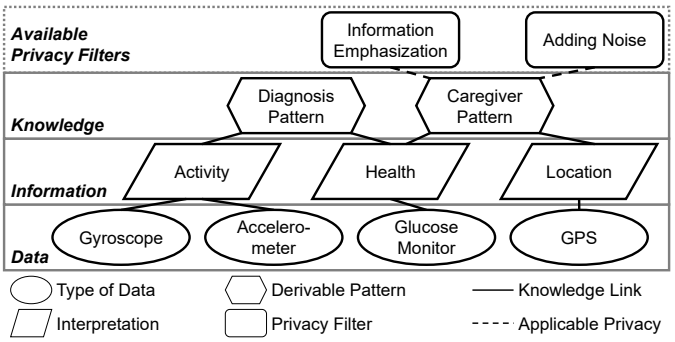


Figure 12. The VAULT Knowledge Model for a Health Record (excerpt).

Based on the Data Layer, the *Information Layer* describes how the available types of data can be interpreted. For instance, GPS data (i. e., latitude, longitude, etc.) can be interpreted as the location of a user. Yet, this is not necessarily a $1:1$ mapping. Certain interpretations are only feasible by combining several types of data. For instance, the activity of a user can only be recognized when both, gyroscope data and accelerometer data, are available. A single type of data can also reveal different kinds of information. For instance, the accelerometer data can also indicate the speed of the user—the "speed" pattern is not modeled in Figure 12 for the sake of simplicity.

At the *Knowledge Layer*, the Smart Services are considered, i. e., which knowledge patterns can be derived when the underlying data are shared with a service. For instance, a diagnosis pattern describes how specific health data change when a user's activity and location are taken into account.

Furthermore, the VAULT knowledge model specifies all available privacy filters, which can be applied to the data sources without concealing a certain knowledge pattern. On the one hand, this takes into account whether the respective filter matches the data type of the source (e. g., a filter for numeric values cannot be applied to free text data) and, on the other hand, whether the data quality after applying the filter is still sufficient to meet the requirements of the services in question.

This enables users to specify a high-level description of their privacy requirements (at the Knowledge Layer) and VAULT is able to identify the appropriate privacy filters and apply them to all associated raw data [63].

Step ⓐ A registered service authenticates to VAULT with its attributes. To prevent a service from getting too many permissions by falsifying its attributes, Gritti, Önen, and Molva [64] introduce a process for verifying these attributes. This approach takes into account that the privacy of the service has to be ensured as well, as the attributes might contain private information about the service provider. This approach is therefore a valuable supplement to the authentication process of a data provisioning platform, such as VAULT [65]. More information on this step is given in Section VI-A.

Step ⓑ If a service is authorized to use VAULT, its queries are temporarily stored in a query buffer.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

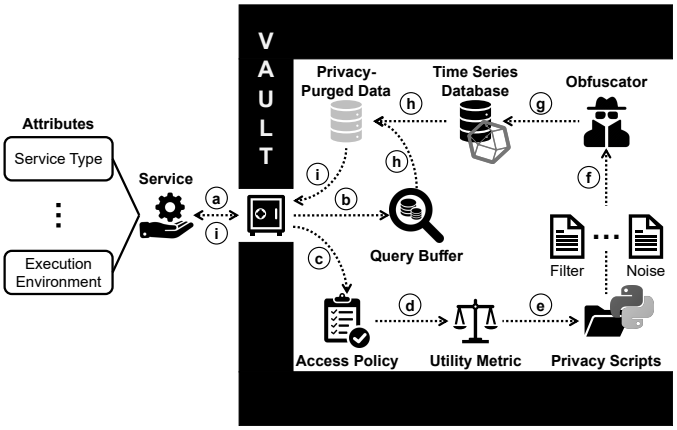


Figure 13. Implementation of and Query Processing in VAULT.

Step ⓒ VAULT checks in the access policy which quality requirements this service has, and which permissions are granted to its attributes.

Step ⓓ Then, a utility metric is used to search for privacy techniques that maximize both, privacy and service quality. Basically, it compares how much information relevant to the service is concealed and how much private data are disclosed when a particular privacy technique is applied. Additionally, the user can determine via a weight, whether his or her focus is more on privacy or service quality [26], [48]. More information on Step ⓒ and ⓓ is given in Section VI-B.

Step ⓔ We implemented each of the privacy techniques presented in Section V as Python scripts. These scripts are made available to VAULT in an archive. Further scripts and thus privacy techniques can be added to the archive to extend the functionality of VAULT. The utility metric selects the most suitable scripts and forwards them to the *Obfuscator*. More information on this step is given in Section VI-C.

Step ⓕ The Obfuscator merges the scripts and adjusts them according to the service.

Step ⓖ It then applies the resulting script to the affected time series data.

Step ⓗ In our prototype, we use InfluxDB. However, due to the database abstraction any other time series database can be used as well. The privacy-purged data are made available in materialized views and the queries stored in the query buffer are executed on them. More information on this step is given in Section VI-D.

Step ⓘ Finally, the database abstraction layer—which, in analogy to the result post-processing strategy, performs a final audit—returns the results to the service.

Without any loss of generality, a time series database is used in VAULT. Yet, VAULT can also be applied to a stream processing system for time series data, such as *Kapacitor* [66]. It is also possible to operate a database and a stream processing system in parallel and combine their results [67].

In the following, we provide additional details on four selected implementation aspects, namely *authentication and access control* (see Section VI-A), *permission management* (see Section VI-B), *privacy filters* (see Section VI-C), and *data management* (see Section VI-D).

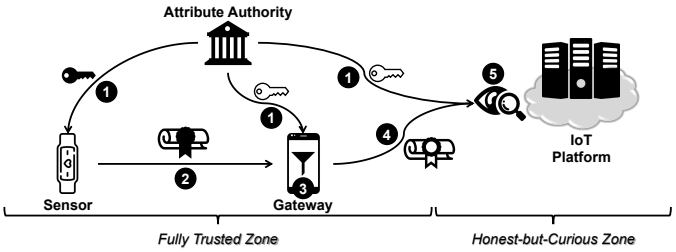


Figure 14. Privacy-Aware Attribute-Based Signature [71].

A. Authentication and Access Control

Gritti, Molva, and Önen [68] introduce a method to identify entities in communication networks by means of their characteristic attributes (e. g., its IP address). This method is based on asymmetric encryption, in which an entity uses a *private key* to sign its messages. All other participants in the network can then verify the authenticity of the messages using the *public key* of that entity.

Although this procedure is already rather lightweight, the verification of the messages (i. e., the decryption of the messages with the public key) still causes costs in terms of computing effort and thus a higher power consumption. This can be a problem especially in a resource-limited environment such as the IoT—in particular for battery-powered IoT devices. It is therefore advisable to outsource a large part of the computing-intensive tasks to a Cloud infrastructure [69].

However, this reveals a lot of private information about the entities in the network towards the Cloud infrastructure, since the public keys are also based on their characteristic attributes. Therefore, privacy must also be considered and preserved in this authentication process [70].

In VAULT, we can make use of the layered architecture of IoT health Smart Services (see Figure 1) for this purpose. In particular, we can rely on the fact that the devices that serve as a gateway to the Big Data Layer generally have sufficient computing power and can therefore handle the computing-intensive tasks effortlessly.

Figure 14 shows how we can apply the approach of Gritti, Önen, and Molva [70] in that kind of IoT environment:

Step ❶ Initially, each entity, i. e., in our context each Smart Device, requires a public and private key. For this purpose, a trusted authority is required. This could be a federal data protection authority for example. This authority verifies the attributes of the entity, generates corresponding key pairs, and distributes the keys to all participants in the network—of course, the private key is only sent to the respective entity itself.

Here we distinguish between *full keys* (depicted in black) and *delegated keys* (depicted in white). If τ is the set of all identifying attributes of an entity, then only a full key reflects all attributes in τ. Just like private keys, full keys are sent only to the entity itself as they contain a lot of information about the entity in question. In contrast, delegated keys contain

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

only a subset of the attributes $\tau' \subsetneq \tau$ and can therefore also be shared with the other participants in the network.

Step ❷ Now, entities can sign their messages with their private key. This not only verifies the authenticity of the messages, but also enables all other participants to immediately notice when the message has been tampered with.

Step ❸ However, this approach not only ensures the authenticity of messages. The gateway has an authentication policy ρ. This policy ρ defines which attributes and attribute values are mandatory for an entity in order to participate in the network. That is, the gateway checks, whether τ satisfy ρ. Only if this is the case, the signature is valid and therefore the message can be considered as having integrity, i. e., it can be approved for forwarding to the network.

Step ❹ Prior to forwarding the message, however, the gateway has to make the signature of the message privacy-aware, i. e., it has to filter out all attributes $\tau \setminus \tau'$. To this end, the gateway uses its delegated key.

Step ❺ Obviously, this reduced signature no longer satisfies the authentication policy ρ. Therefore, an additional authentication policy ρ' is required that corresponds to τ'. Using this reduced authentication policy ρ' any participant of the network can verify that the gateway has approved the message, i. e., they are still able to verify the authenticity of a message without gaining access to the full set of the sender's attributes τ.

For more information on this process, please refer to literature [64], [68]–[71].

In VAULT, we apply this approach to identify Smart Services. A Smart Service must sign each of its queries with its identifying attributes. The access control can then check the signature and use the authentication policy to verify which attribute values the service currently has. Such attribute values can include among others the type of service or its execution environment. That way, attribute-based access control is enabled. In a dynamic IoT scenario, as addressed in VAULT, an attribute-based access control is particularly suitable [72].

In the following, we take a closer look at how the permission management in VAULT is organized, which is based on this attribute-based access control.

B. Permission Management

In VAULT, we apply a context-based permission model. Such a model is especially appropriate for use in an IoT scenario. On the one hand, the inclusion of context data allows to specify permission rules in a far more flexible manner. On the other hand, IoT devices capture a lot of context information anyway. This information can therefore be used for the purpose of assigning permissions at no additional cost [73].

Figure 15 shows the permission model used in VAULT. As shown in the figure, a VAULT policy rule consists of four key components: the *data* that are to be accessed, the entity that requests access to them (i. e., the *Smart Service* in question), the *context* under which the access is to take place, and the privacy *constraints* that apply to this access (i. e., which privacy filters have to be applied to the data).

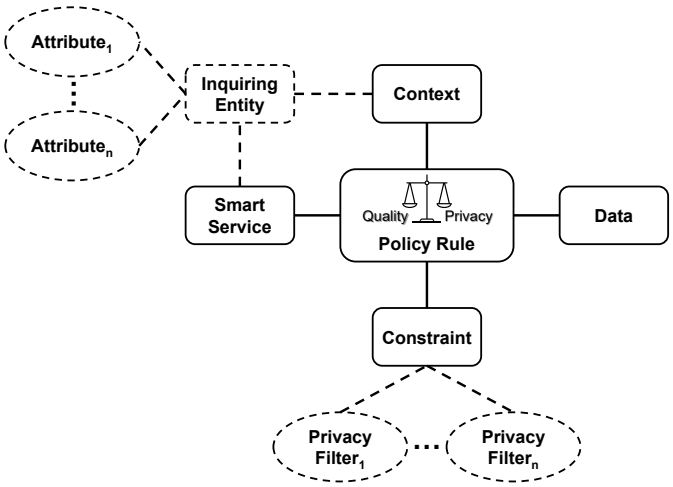


Figure 15. Permission Model Applied in VAULT.

The data component represents an abstraction of the VAULT knowledge model (see Figure 12). That is, a user can specify his or her privacy requirements at any layer of the knowledge model (data, information, or knowledge). Due to the knowledge links modeled in the knowledge model, VAULT is able to derive policy rules at a data level. In other words, the privacy requirements are mapped to the affected data sources.

The information required for the Smart Service component is gathered by VAULT during authentication. Since a Smart Service has to sign all data requests with its characteristic attributes, the originator of each request can be uniquely identified.

In addition, the current attribute values of an inquiring entity are also checked during authentication. These values give an indication about the context in which a data request is made. Using this context, a user can specify, for instance, the purpose for which a request has to be granted. The GDPR explicitly states in Article 9(2) that the processing of highly sensitive data such as health data is only permissible if the data subject has given his or her explicit consent. The specified purposes must also be specified in this regard.

With these three components binary permissions—access is either granted or denied—could be specified. However, in the VAULT permission model fine-grained permissions are envisaged. To this end, the constraint component specifies which privacy filters should be applied to the data before they are shared with the Smart Service. More information on how such a privacy filter operates can be found in Section VI-C.

VAULT adopts a *Privacy by Default* approach as proposed by the GDPR in Article 25. In the context of VAULT, this means that data can only be accessed if a user has specified a respective policy rule.

Besides the privacy requirements of the users, the VAULT permission model also considers the quality requirements of the Smart Services. On the one hand, the knowledge model excludes inappropriate privacy filters, i. e., filters that either do not match the data types in question or that impair the data quality to such an extent that they become useless for the

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

VAULT \ Original Data	(+)	(−)
(+)	TP	FP
(−)	FN	TN

Figure 16. Confusion Matrix for the Utility Metric Applied in VAULT.

corresponding Smart Service. On the other hand, the permission model applies a utility metric to select the filter that provides an optimal balance between privacy and quality.

For this metric, a confusion matrix is used. Figure 16 shows such a confusion matrix. To put it simply, the matrix compares how accurately a Smart Service operates on the data manipulated by the VAULT privacy filters as opposed to the original data. In other words, it determines how often a Smart Service operates identically on both data sets (true positives TP and true negatives TN), how often it fails to detect a pattern in the manipulated data (false negatives FN), and how often it incorrectly detects a pattern in the manipulated data (false positives FP).

These measures can then be used to calculate metrics similar to those found in the field of machine learning. For instance, *accuracy* describes the closeness of the measurements to a specific value:

$$ACC = \frac{TP+TN}{TP+TN+FP+FN}$$

Another useful metric is the *F1 score*, which calculates the weighted harmonic mean of precision and recall:

$$F_1 = \frac{2*TP}{2*TP+FP+FN}$$

Moreover, penalty weights can be assigned to these measures, for instance if a false negative is considered worse in a certain use case than a false positive [26], [48]. In our AAL scenario, e. g., it is not as critical if a caregiver is falsely informed an accidental fall of a helpless person as when s/he is not notified about an actual fall.

Using these metrics, VAULT is able to select an appropriate privacy filter. One of these filters is discussed in detail hereafter.

C. Privacy Filters

In the field of privacy preserving techniques, adding noise to data is of special importance. On the one hand, this procedure is very straightforward in terms of computational effort and complexity, and on the other hand, it still conceals data reliably. It also constitutes the foundation for many other techniques, such as differential privacy [50]–[52]. Hence, out of the five techniques presented in Section V, we focus on noise-based privacy filters for VAULT in the following.

Listing 3 shows the Python code for the most basic noise-based privacy filter possible. In it, Gaussian noise is added to a data series. Gaussian noise generates noise that has a normal distribution, i. e., corresponds to the following probability density function:

$$r(x) = \frac{1}{\sigma * \sqrt{2*\pi}} * e^{-\frac{1}{2}*(\frac{x-\mu}{\sigma})^2}$$

μ is the mean of the distribution, while σ is its standard deviation (respectively, σ^2 is its variance).

With the default parameters $\mu = 0.075$ and $\sigma = 0.35$, we created the noisy data shown in Figure 10 with this privacy filter. At first sight, it seems that this prevents any detail from being revealed from these noisy data. However, since these data are time series data, techniques from the field of signal processing can be applied to them [74].

One of these techniques is the *Discrete Wavelet Transform*. Figure 18a illustrates this technique. In the field of signal processing wavelet transforms are used for the compression of signals. A signal (or in our case time series data) of length n is split into two coefficient series each of length $n/2$ using high-pass filters and low-pass filters. Similar to the Continuous Wavelet Transform a mother wavelet is used to this end.

With each split, the high-pass filter provides detailed coefficients for the respective frequency band, while the low-pass filter provides approximating coefficients, which are further split in subsequent steps. Thereby, a Discrete Wavelet Transform splits the signal into $\lceil log_2(n) \rceil$ frequency bands[2].

The result of a Discrete Wavelet Transform for the raw data and the noisy data from the example given in Figure 10 is presented in Figure 17a and Figure 17b. For this transform, the *Haar Wavelet* was applied as a mother wavelet.

It can be seen that the raw data and the noisy data differ only in the two uppermost frequency bands (L1 and L2). The remaining frequency bands are almost completely unaffected by the noise. In addition, the underlying course on frequency band L2 can still be identified quite distinctly.

Therefore, a noise filter can be applied to these two frequency bands L1 and L2 of the noisy data to come very close to the frequency bands of the raw data. Using the Inverse Wavelet Transform, as shown in Figure 18b, the frequency bands can then be merged back into a single signal. To this end, the detailed coefficients and the approximating coefficients are iteratively joined.

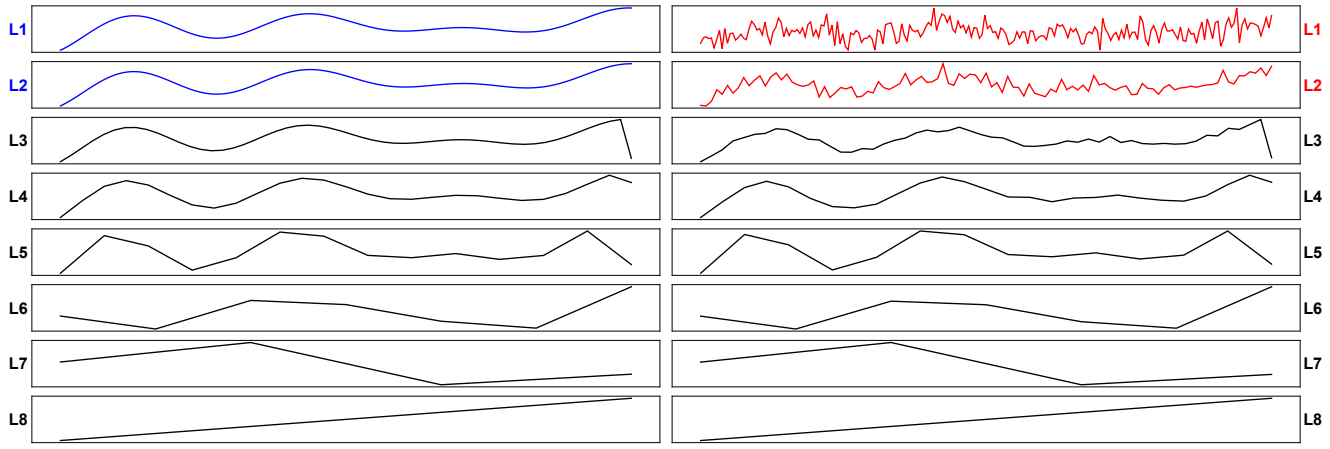
The *Savitzky-Golay filter* is a digital filter which is often used in the field of time series data for smoothing the data

[2]In each iteration, the number of coefficients is bisected, whereby a complete Discrete Wavelet Transform results in $\lceil log_2(n) \rceil$ discrete frequency bands.

```
1 import numpy as np
2
3 def add_noise(data : np.array, mu : float = 0.075,
  ↪  sigma : float = 0.35):
4     noise = np.random.normal(mu, sigma, len(data))
5     return data + noise
```

Listing 3. Simple Noise-Based VAULT Privacy Filter.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(a) Frequency Bands of the Raw Data.

(b) Frequency Bands of the Noisy Data.

Figure 17. Application of a Discrete Wavelet Transform to the Data Series Shown in Figure 10.

series. This filter is not only very effective, but also highly efficient for the denoising of time series data [75].

In Algorithm 1, a method is shown how to use Discrete Wavelet Transform and the Savitzky-Golay filter to remove artificially generated noise. For this purpose, a time series is first decomposed into frequency bands, then the Savitzky-Golay filter is applied to the uppermost $\lceil log_2(n)/4 \rceil$ frequency bands, and finally the time series is reconstructed using Inverse Wavelet Transform.

Figure 19 shows the resulting time series when Algorithm 1 is applied to the noisy data given in Figure 10. As is evident when comparing the two data series, the noise can be almost completely removed, i. e., an almost lossless reconstruction of the original data is possible.

Since this renders the privacy filter given in Listing 3 virtually useless, an improved method is applied in VAULT. This improved noise technique is based on the *SNIL Algorithm*. SNIL stands for "spread noise to intermediate levels of wavelet coefficients". This algorithm generates resilient noise in time series data without impairing the data quality more than simply adding Gaussian noise to the data [76], [77].

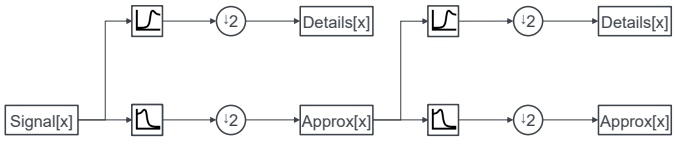
Algorithm 2 shows how the SNIL Algorithm operates. First, the original data series is decomposed into frequency bands using Discrete Wavelet Transform. Then, Gaussian noise is added to all frequency bands between an initial level l_s and an end level l_e. Finally, all frequency bands are merged again using Inverse Wavelet Transform. The resulting data series is then made available to the respective Smart Services by VAULT.

Our experiments have shown that $l_s = \lfloor log_2(n)/4 \rfloor$ and $l_e = \lceil log_2(n)/2 \rceil + 1$ leads to the best possible result. However, this can be freely adjusted if in other use cases this parameterization either offers too little privacy protection or if the data quality deteriorates too much.

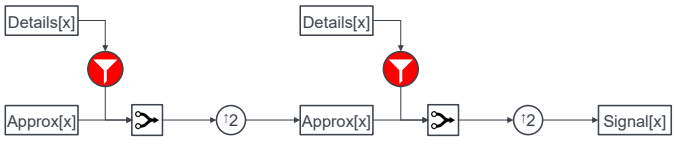
Figure 20 shows the frequency bands of the time series from our running example after applying the SNIL Algorithm. Gaussian noise is added to the frequency bands L2 to L5. The resulting noisy time series is shown in Figure 21. As it can be seen, the utility of the data is preserved—with regard to the course of the blood glucose curve—despite the resilient noise.

Listing 4 shows how the SNIL-based privacy filter is implemented in VAULT. In it, we increase the parameter σ, i. e., the standard deviation, per frequency level. The wavelet transform is performed by the *PyWavelets* module [78], [79].

In a similar way, VAULT implements further profound noise-based privacy filters as well as filters for the other privacy techniques presented in Section V. The application of these privacy filters, however, results in many different privacy-aware variants of each time series. How all these data are managed by VAULT is described hereafter.



(a) Discrete Wavelet Transform.



(b) Inverse Wavelet Transform.

Figure 18. Illustration of the Wavelet Transform Process.

Algorithm 1: Savitzky-Golay-Based Denoising.

input : noisy time series data *data*
output : denoised time series

```
freq ← decomposition of data into frequency bands;
for i ← 1 to ⌈len(freq)/4⌉ do
    apply Savitzky-Golay filter to frequency level i;
end
clean ← merge of filtered frequency bands;
return clean;
```

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

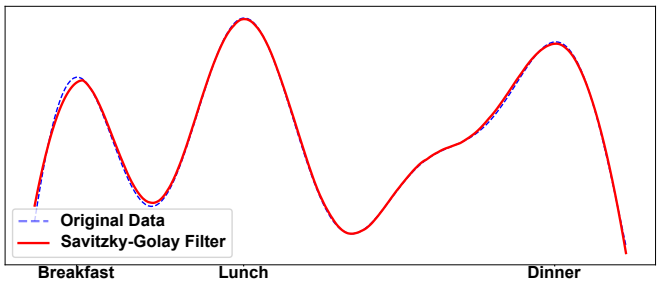


Figure 19. Reconstructed Time Series Using Algorithm 1.

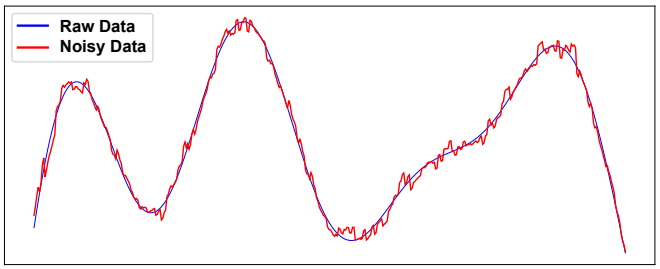


Figure 21. Adding SNIL-Based Noise to Time Series Data.

D. Data Management

Due to the increasing digitization in all areas of life enabled by the IoT, data management systems and data analytics systems are facing entirely new challenges when dealing with *big data*. Big data are characterized by four Vs: *volume*, *variety*, *velocity*, and *veracity*. A data management system that is designed to provide Smart Services with a data foundation for decision-making has therefore not only to be able to handle very large volumes of heterogeneous data that are generated constantly (and should be available to services in *near real-time*), but it has also to ensure that the data quality of the provided data is as good as possible. Since traditional data management architectures, such as *Data Warehouses*, cannot cope with these new big data challenges, *Data Lakes* introduce a completely new approach [80].

As Data Lakes are initially only a general concept that requires a concrete implementation, many different perceptions of this concept exist. Basically, the concept behind a data lake is that it stores all acquired data, even if there is currently no use for it. In addition, the stored data should be pre-processed (e. g., by *data wrangling*) in order to increase its quality and to be able to process queries more efficiently. Therefore, it can be considered as a general consensus that a Data Lake has to be flexible in order to support any kind of use case and has to be able to provide original raw data in addition to pre-processed data—that way, the obligation to produce proof can be fulfilled [81].

There is also no agreement on the internal structure of such a data lake. However, the zone models have now become widely

Algorithm 2: Creating SNIL-Based Noise.

input : time series data *data*; start frequency level l_s; end frequency level l_e
output : time series with filter-resistant noise

```
freq ← decomposition of data into frequency bands;
for i ← l_s to l_e do
    foreach coefficient c in freq[i] do
        add Gaussian noise to c;
    end
end
noisy ← merge of partially noisy frequency bands;
return noisy;
```

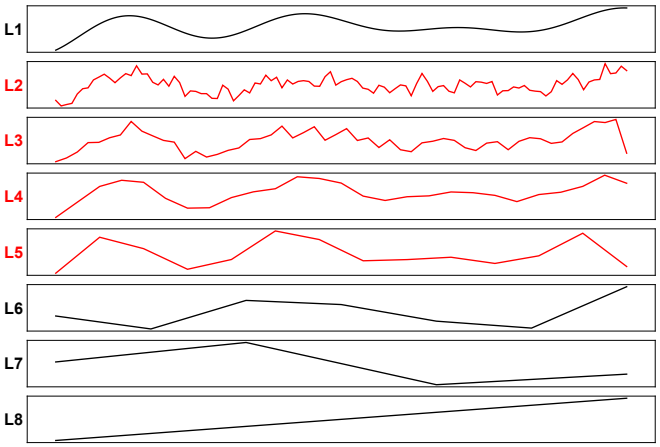


Figure 20. Impact of the SNIL Algorithm on a Time Series.

```
import pywt

""" selecting the Haar mother wavelet and    """
""" determine the number of frequency levels """
w = pywt.Wavelet('haar')
maxlev = pywt.dwt_max_level(len(data), w.dec_len)

""" coefficients of each frequency level """
fl = []

""" stepwise decomposition of the data """
for i in range(maxlev):
    (cA, cD) = pywt.dwt(cA, w, 'periodic')
    fl.append(cD)

""" adding noise to the frequency levels """
for i in range(maxlev // 4 - 1, maxlev // 2 + 1):
    s = i * 0.35
    noise = np.random.normal(0, s, len(fl[i]))
    fl[i] = fl[i] + noise

""" inverse wavelet transform and adjustment"""
noisy_data = cA
for i in range(maxlev):
    if (len(noisy_data) < len(fl[maxlev-1-i])):
        fl[maxlev - 1 - i] =
         ↪  fl[maxlev-1-i][:len(noisy_data)]
    if (len(noisy_data) > len(fl[maxlev-1-i])):
        noisy_data =
         ↪  noisy_data[:len(fl[maxlev-1-i])]
    noisy_data = pywt.idwt(noisy_data,
     ↪  fl[maxlev-1-i], w, 'periodic')
```

Listing 4. SNIL-Based VAULT Privacy Filter.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

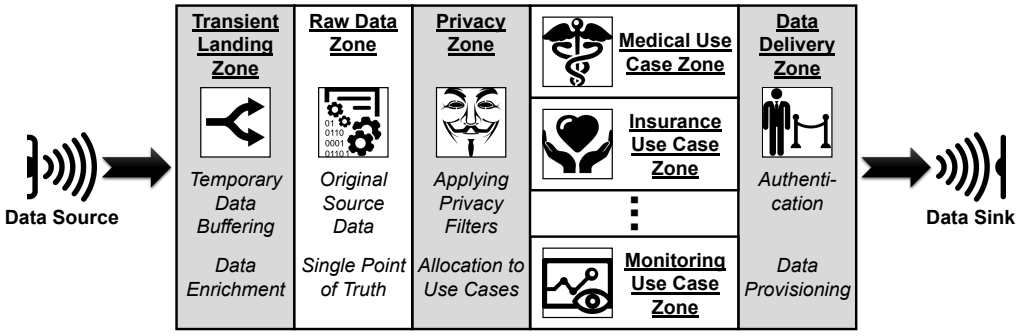


Figure 22. A Multi-Zone Data Lake Architecture for the Data Management in VAULT.

accepted. In these models there are several zones defined that contain the data stored in the data lake. Each zone is described by the respective degree of how the data in it is pre-processed, e. g., "raw", "cleansed", or "aggregated". Applications such as Smart Services can thereby select the most suitable data processing level for their purposes via the zones. While most of these zones are rather generic and thus open for any kind of application, special processing techniques can also be used, which are only relevant for a few selected use cases [82].

In VAULT, similar problems have to be solved. On the one hand, the IoT context requires the handling of big data and the associated four Vs in terms of data management. On the other hand, the gathered data has to be pre-processed in accordance with the privacy requirements and made available on-demand in accordance with the quality requirements.

To this end, we introduce a multi-zone data lake architecture for VAULT. In this architecture data sources (e. g., IoT devices) are isolated from data sinks (e. g., Smart Services). The data management concept behind our architecture is based on the zone model for data lakes by Sharma [83].

This architecture is shown in Figure 22. Grey zones are *processing zones*—i. e., data are only passed through and preprocessed for the subsequent zones—while white zones are *storage zones*—i. e., data are stored there permanently and are made available for further processing.

Incoming data from any kind of data source such as relational data stores, sensors, and social media arrive in the *Transient Landing Zone*. Here, data are temporarily buffered and enriched with metadata. For instance, it can be annotated from which sensor the data was captured or in which units the measurements are presented. An initial data purging can also be performed in this zone.

If, e. g., one of the IoT devices transmits unusually high values while all other devices monitoring the same parameters have not registered any abnormal values, this zone tags these values as suspicious[3]. Furthermore, the data can also be split up in this zone. For instance, if the payload of an IoT device is composed of several measurements, it is useful to treat them as individual measurements in the subsequent zones.

The enriched data are then transferred to the *Raw Data Zone*. The raw data are stored there unmodified[4] as originally received from the sources. The Raw Data Zone is the single source of truth for all subsequent zones. Smart Services have no direct access to data in this zone. Only internal accesses are permitted.

In order to provide data access to Smart Services, data have to be propagated to one of the horizontal *Use Case Zones*[5]. In our AAL application scenario (see Section II), for instance, the *Medical Use Case Zone* would contain all available captured data almost unmodified. Smart Services of physicians are thereby able to make diagnoses correctly. In the *Insurance Use Case Zone*, all information would be available so that an insurance company could check, for instance, whether a person is performing health-promoting measures (e. g., sports exercises) and thus qualifies for a bonus program (i. e., a lower insurance premium). However, details on health data are not available here. Finally, the *Monitoring Use Case Zone* contains all data required by a care provider for remote monitoring (e. g., alerting in case of a fall). However, these data are highly distorted so that no unnecessary details are disclosed.

These zones are appropriately populated by the *Privacy Zone*. In the Privacy Zone, one or more privacy filters can be applied to the data (see Section VI-C) according to the specifications in the knowledge model (see Figure 12).

Smart Services cannot access data directly, but only via the *Data Delivery Zone*. This zone operates as an access control layer. On the one hand, the verification of the signatures (i. e., the identification of the Smart Services, see Section VI-A) is done in this zone. On the other hand, VAULT's permission model is used to select the Use Case Zone to which the respective Smart Service should have access (see Section VI-B).

Due to this architecture it is possible to efficiently manage the big data handled by VAULT and to provide the Smart Services with the data they need while still taking privacy and quality requirements into account. Moreover, all data protection

[3] A premature removal of such values is not intended as it cannot be guaranteed that the detected anomaly is not a correct measurement.

[4] "Unmodified" refers to the data quality and the data format. The previous enrichment with metadata as well as the splitting into individual measurements is of course retained.

[5] Vertical zones affect all data, while horizontal zones affect only a subset of the data.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

concepts of VAULT integrate seamlessly into this architecture, which further strengthens our Privacy by Default philosophy.

VII. Evaluation

Having presented VAULT's concept and implementation, we now evaluate its effectiveness and efficiency. To this end, we first discuss whether it meets the requirements towards a privacy system for Smart Services (see Section II) in Section VII-A. Then, we carry out a performance analysis for three selected privacy filters in Section VII-B.

A. Assessment

In VAULT, each user is able to specify his or her individual privacy requirements. Since this is done in natural language and the mapping to actual data sources can be realized automatically, the configuration is also user-friendly. That way, users are enabled to specify their privacy requirements very precisely and VAULT fulfills these requirements as good as possible ($\mathbf{R_1}$).

VAULT also preserves the utility of a service when it is compatible with the privacy requirements. This is made possible by the specification of the service's quality requirements. This ensures that the service receives usable data in terms of quantity and quality. That is not the case with approaches working only with data suppression or mock data, which have a sustainably negative impact on these two parameters ($\mathbf{R_2}$).

The utility metric applied in VAULT balances privacy and quality requirements against each other and determines the best configuration. It aims to maximize both, the amount of concealed private data as well as the amount of revealed information, which is relevant to the service. As it might not be possible to maximize both of these values at the same time, at least Pareto optimality is achieved. The user can also weight, which of these objectives should be preferred by VAULT ($\mathbf{R_3}$).

To this end, VAULT provides five different privacy techniques that are tailored to IoT time series data. Each of these techniques deals with different privacy aspects. Furthermore, these techniques can be extended and combined so that a suitable technique can be found for every use case ($\mathbf{R_4}$).

In VAULT, permissions (and thus the applied privacy techniques) are not assigned to a service, but to a specific combination of its attributes. This enables a considerably more dynamic permission assignment ($\mathbf{R_5}$).

Table II. Characteristics of the Data Set Used for the Evaluation.

Metric	Value
Number of Data Records	8,192,000
Data Volume	73.9 MB
Distinct Data Records	137
Mean	127.7
Standard Deviation	22.9
Minimum	101.0
Lower Percentile	111.0
Median	119.0
Upper Percentile	138.0
Maximum	256.0

Thus, VAULT fulfills all requirements towards a privacy system for time series data as processed by Smart Services.

B. Performance Analysis

To evaluate the efficiency of VAULT's privacy filters, we have generated an artificial blood glucose data set. This data set consists of more than 8 million individual blood glucose readings. This results in a data volume of over 70 MB. Table II summarizes the relevant metrics for the data set.

For the performance analysis, we measured how long three privacy filters, namely a cubic-spline-based filter, a Savitzky-Golay-based filter, and a SNIL-based filter (as representatives of *data interpolation*, *data smoothing*, and *adding noise*—see Section V) need to process the data. These filters are chosen as they cause the highest computational effort. They are implemented using Python 3.9.0, NumPy 1.19.0, pandas 1.1.4, SciPy 1.5.4, and PyWavelets 1.1.1.

For the measurements, we initially generated 15 subsets of our original data set, by progressively halving the number of contained data records—i. e., we ended up with 15 data sets containing between 500 and 8,192,000 data records. We processed each of these data sets 10 times with each of the three privacy filters, measured the required computing time, and calculated the arithmetic mean to mitigate outliers. These measurements were executed on a standard desktop computer (Intel Core i7-8700 processor, 16 GB of main memory). The results are shown in Figure 23.

Generally, the overhead caused by our filters is very low. An increase in computing time can only be observed for 512,000 data records and above, i. e., a data volume of at least 4.61 MB. If the number of data records is lower, only a negligible basic workload is observable. Only the cubic-spline-based filter has a slightly higher basic workload since the private data points that have to be concealed must first be identified[6].

For larger data sets, i. e., data sets that include more data records, the increased computing time is noticeable. For all three filters, however, this performance overhead increases linearly to the number of data records that have to be processed[7]. The measured computing times are shown in Table III. These numbers imply that all three filters have a runtime of $\mathcal{O}(n)$.

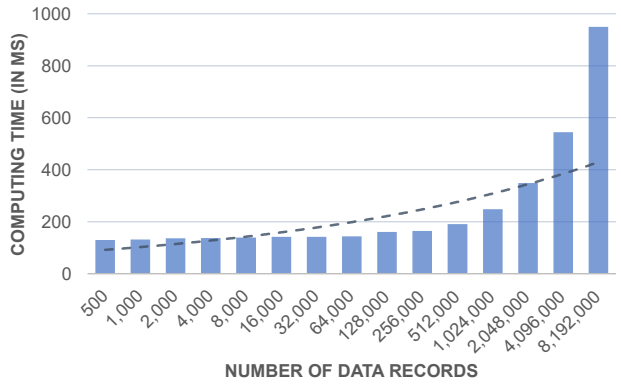
This overhead can also be regarded as runtime overhead for VAULT as a whole (in comparison to a data provision without privacy features) since the privacy filters represent its

[6]In our performance analysis, we classified 1 % of the data points as private.
[7]Note that in Figure 23, we use a logarithmic scale for the x-axis.

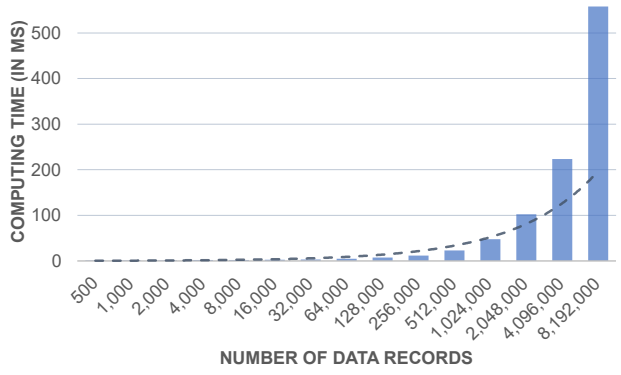
Table III. Performance Analysis Results Overview.

	Cubic Spline	Savitzky-Golay	SNIL
Mean Basic Workload	142.7 ms	3.6 ms	2.0 ms
512,000 Records	190.8 ms	22.9 ms	12.9 ms
1,024,000 Records	248.4 ms	47.9 ms	23.4 ms
2,048,000 Records	349.0 ms	102.1 ms	41.8 ms
4,096,000 Records	544.2 ms	223.5 ms	83.1 ms
8,192,000 Records	949.5 ms	558.2 ms	158.4 ms

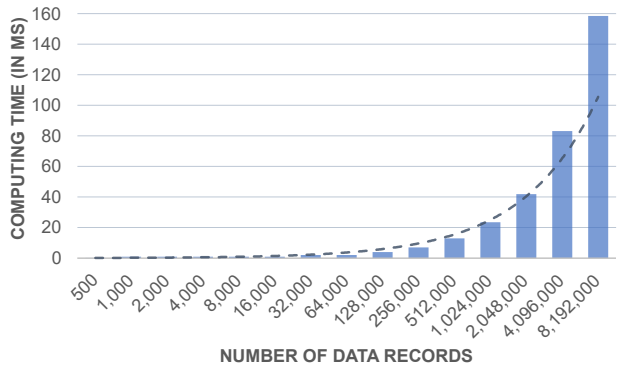
2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(a) Runtime Overhead Caused by a Cubic-Spline-Based Filter.



(b) Runtime Overhead Caused by a Savitzky-Golay-Based Filter.



(c) Runtime Overhead Caused by a SNIL-Based Filter.

Figure 23. Computing Time Analysis of the Privacy Filters in VAULT.

most complex component in terms of computational effort. So, VAULT is efficient—even for large data sets, filtering at runtime is feasible. Materialized views (and the consequent higher storage requirements) are only needed for a vast amount of data or particularly complex privacy filters.

VIII. Conclusion and Future Work

The tremendous progress that IoT-enabled devices have made in recent years in terms of computing power, transmission speed, and sensor technology provides the technical foundation for a wide range of IoT applications. Such Smart Services affect all aspects of our daily lives (e. g., Smart Homes, Smart Cars, and Smart Health). In order to enjoy the benefits of these services, however, users have to disclose a lot of data, some of which revealing highly sensitive information. However, current privacy approaches are not adapted to the specific characteristics of time series data as processed by Smart Services, making them unnecessarily restrictive. As a result, users have to disclose too much private information in order to prevent that the service quality deteriorates too much.

In this paper, we therefore introduce VAULT, a new privacy concept for time series data. In VAULT, IoT data are managed and provided to Smart Services on demand. In this process, VAULT not only considers the privacy requirements of the users but also the quality requirements of the services in order to achieve a high level of data utility. To ensure this, VAULT introduces four important building blocks: *A.* An IoT-enabled *authentication* mechanism ensures that Smart Services can be identified via their characteristic attributes. In addition, this mechanism can also verify their current attribute values, e. g., in which country they are currently hosted. This allows VAULT to apply an attribute-based *access control* for its managed data. This facilitates fine-grained access rules. *B.* These rules are defined in VAULT's permission model. The model not only takes into account which Smart Service wants to access the data, but also the current context of that service. Furthermore, additional restrictions can be specified whether and how the data has to be distorted prior to being shared. The *permission management* in VAULT uses a utility metric that evaluates how much the data quality suffers from the use of a certain privacy technique. *C.* VAULT introduced five different concepts for such privacy techniques, which are tailored to the special characteristics of time series data. Each of these concepts is able to conceal a different kind of private information in the data. For instance, projection, selection, and information emphasis are suitable for data reduction, whereas data interpolation and data smoothing can be used as noise filters or for outlier suppression. Thus, VAULT can find a good ratio between privacy and service quality. Different implementations of these techniques are deployed in VAULT in the form of Python scripts, called *privacy filters*. *D.* These three building blocks are combined in a multi-zone architecture for the management of big data. This architecture not only enables an efficient *data management* but is also the prerequisite for the provisioning of high-utility time series data to Smart Services. These features render VAULT a Privacy by Design data management and provisioning solution for the IoT as required by GDPR.

As part of future work, we aim to further *evaluate the performance* of VAULT in terms of data throughput. As the performance of such a system highly depends on the data being processed, a conclusive evaluation has to be based on real-world data. This is not feasible for medical data due to the high restrictions such data involve. So, the evaluation will be based on data from the food chemistry domain [84]. A possible application scenario could be the provisioning of end-to-end data about the food production chain (e. g., allergens a food product had contact with during production) [85]. Yet, the evaluation results based on artificial data are very promising.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Another aim is to improve the performance of VAULT. Our research in the field of data lakes has revealed that a sophisticated *metadata management* facilitated information retrieval [86]. In this way, we can increase the overall performance of VAULT as well.

The quality of VAULT can also be improved. For instance, better purging filters can be used in the Transient Landing Zone. Litou *et al.* [87], [88] introduce a method to detect *misinformation*. By applying this method, VAULT could tag such data as incorrect at an early stage, which improves the veracity of the data stock.

The trust in the data provided by VAULT can be further enhanced. Due to the applied authentication mechanism, the authenticity of the data and their provenance can be verified. Yet, attackers could manipulate these data after they have been stored in VAULT. By integrating immutable and tamper-resistant blockchain technologies in the Raw Data Zone, an *end-to-end data authentication* can be achieved [89]. To ensure efficient access to these data, novel access structures for such data storages are required [90].

Acknowledgment

Parts of this work are results of the research projects PATRON commissioned by the Baden-Württemberg Stiftung gGmbH and DiStOPT (252975529) funded by the DFG.

References

[1] C. Stach, "VAULT: A Privacy Approach towards High-Utility Time Series Data," in *Proceedings of the Thirteenth International Conference on Emerging Security Information, Systems and Technologies*, series SECURWARE '19, 2019, pages 41–46.

[2] S. Dreyer, D. Olivotti, B. Lebek, and M. H. Breitner, "Focusing the customer through smart services: A literature review," *Electron Markets*, volume 29, pages 55–78, 2019. DOI: 10.1007/s12525-019-00328-z.

[3] D. Marikyan, S. Papagiannidis, and E. Alamanos, "A systematic review of the smart home literature: A user perspective," *Technological Forecasting and Social Change*, volume 138, pages 139–154, 2019. DOI: 10.1016/j.techfore.2018.08.015.

[4] F. Zantalis, G. Koulouras, S. Karabetsos, and D. Kandris, "A Review of Machine Learning and IoT in Smart Transportation," *Future Internet*, volume 11, number 4, 94:1–94:23, 2019. DOI: 10.3390/fi11040094.

[5] P. Panchatcharam and Vivekanandan S., "Internet of Things (IOT) in Healthcare – Smart Health and Surveillance, Architectures, Security Analysis and Data Transfer: A Review," *International Journal of Software Innovation*, volume 7, number 2, pages 21–40, 2020. DOI: 10.4018/IJSI.2019040103.

[6] M. S. Jalali, J. P. Kaiser, M. Siegel, and S. Madnick, "The Internet of Things Promises New Benefits and Risks: A Systematic Analysis of Adoption Dynamics of IoT Products," *IEEE Security Privacy*, volume 17, number 2, pages 39–48, 2019. DOI: 10.1109/MSEC.2018.2888780.

[7] Q. Pan, "Privacy in the New Age of IoT," in *Women Securing the Future with TIPPSS for IoT*, F. D. Hudson, Ed. Springer, 2019, pages 37–52. DOI: 10.1007/978-3-030-15705-0_3.

[8] European Parliament and Council of the European Union, "Regulation on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/EC," Legislative acts L119, 2016.

[9] S. Wachter, "Normative Challenges of Identification in the Internet of Things: Privacy, Profiling, Discrimination, and the GDPR," *Computer Law & Security Review*, volume 34, number 3, pages 436–449, 2018. DOI: 10.2139/ssrn.3083554.

[10] K. M. Ramokapane, A. C. Mazeli, and A. Rashid, "Skip, Skip, Skip, Accept!!!: A Study on the Usability of Smartphone Manufacturer Provided Default Features and User Privacy," *Proceedings on Privacy Enhancing Technologies*, volume 2019, number 2, pages 209–227, 2019. DOI: 10.2478/popets-2019-0027.

[11] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner, "Android Permissions: User Attention, Comprehension, and Behavior," in *Proceedings of the Eighth Symposium on Usable Privacy and Security*, series SOUPS '12, 2012, 3:1–3:14. DOI: 10.1145/2335356.2335360.

[12] D. Akhawe and A. P. Felt, "Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness," in *Proceedings of the 22nd USENIX Conference on Security*, series SEC '13, 2013, pages 257–272. DOI: 10.5555/2534766.2534789.

[13] A. P. Felt, S. Egelman, and D. Wagner, "I've Got 99 Problems, but Vibration Ain't One: A Survey of Smartphone Users' Concerns," in *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, series SPSM '12, 2012, pages 33–44. DOI: 10.1145/2381934.2381943.

[14] R. Chow, "The Last Mile for IoT Privacy," *IEEE Security & Privacy*, volume 15, number 6, pages 73–76, 2017. DOI: 10.1109/MSP.2017.4251118.

[15] InfluxData Inc. (2020). "InfluxDB," [Online]. Available: https://www.influxdata.com/products/influxdb-overview/.

[16] C. Stach *et al.*, "The AVARE PATRON - A Holistic Privacy Approach for the Internet of Things," in *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications*, series SECRYPT '18, volume 2, 2018, pages 372–379. DOI: 10.5220/0006850305380545.

[17] D. Siewiorek, "Generation Smartphone," *IEEE Spectrum*, volume 49, number 9, pages 54–58, 2012. DOI: 10.1109/MSPEC.2012.6281134.

[18] L. Hassan, A. Dias, and J. Hamari, "How motivational feedback increases user's benefits and continued use: A study on gamification, quantified-self and social networking," *International Journal of Information Management*, volume 46, pages 151–162, 2019. DOI: 10.1016/j.ijinfomgt.2018.12.004.

[19] C. Baxter, J.-A. Carroll, B. Keogh, and C. Vandelanotte, "Assessment of Mobile Health Apps Using Built-In Smartphone Sensors for Diagnosis and Treatment: Systematic Survey of Apps Listed in International Curated Health App Libraries," *JMIR mHealth and uHealth*, volume 8, number 2, 16741:1–16741:11, 2020. DOI: 10.2196/16741.

[20] S. K. Vashist and J. H. T. Luong, "Commercially Available Smartphone-Based Personalized Mobile Healthcare Technologies," in *Point-of-Care Technologies Enabling Next-Generation Healthcare Monitoring and Management*. Springer, 2019, pages 81–115. DOI: 10.1007/978-3-030-11416-9_3.

[21] M. Bitsaki *et al.*, "ChronicOnline: Implementing a mHealth solution for monitoring and early alerting in chronic obstructive pulmonary disease," *Health Informatics Journal*, volume 23, number 3, pages 197–207, 2016. DOI: 10.1177/1460458216641480.

[22] F. Steimle, M. Wieland, B. Mitschang, S. Wagner, and F. Leymann, "Extended provisioning, security and analysis techniques for the ECHO health data management system," *Computing*, volume 99, pages 183–201, 2017. DOI: 10.1007/s00607-016-0523-8.

[23] N. Karisalmi, J. Kaipio, and S. Kujala, "Encouraging the Use of eHealth Services: A Survey of Patients' Experiences," *Studies in Health Technology and Informatics*, volume 257, pages 206–211, 2019.

[24] A. M. Ronchi, "e-Health: Background, Today's Implementation and Future Trends," in *e-Services*. Springer, 2019, pages 1–68. DOI: 10.1007/978-3-030-01842-9_1.

[25] M. Knöll, M. Moar, S. Boyd Davis, and M. Saunders, "Spontaneous Interventions for Health: How Digital Games May Supplement Urban Design Projects," in *Technologies of Inclusive Well-Being*, A. L. Brooks, S. Brahnam, and L. C. Jain, Eds. Springer, 2014, pages 245–259. DOI: 10.1007/978-3-642-45432-5_12.

[26] C. Stach, F. Dürr, K. Mindermann, S. M. Palanisamy, and S. Wagner, "How a Pattern-based Privacy System Contributes to Improve Context Recognition," in *Proceedings of the 2018 IEEE International*

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Conference on Pervasive Computing and Communications Workshops, series CoMoRea '18, 2018, pages 238–243. DOI: 10.1109/PERCOMW.2018.8480227.

[27] G. Mincolelli, S. Imbesi, and M. Marchi, "Design for the Active Ageing and Autonomy: The Role of Industrial Design in the Development of the "Habitat" IoT Project," in *Proceedings of the 8th International Conference on Applied Human Factors and Ergonomics*, series AHFE '17, 2017, pages 88–97. DOI: 10.1007/978-3-319-60597-5_8.

[28] A. Dohr, R. Modre-Osprian, M. Drobics, D. Hayn, and G. Schreier, "The Internet of Things for Ambient Assisted Living," in *Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations*, series ITNG '10, 2010, pages 804–809. DOI: 10.1109/ITNG.2010.104.

[29] G. Wang, X. Wang, J. Nie, and L. Lin, "Magnetic-Based Indoor Localization Using Smartphone via a Fusion Algorithm," *IEEE Sensors Journal*, volume 19, number 15, pages 6477–6485, 2019. DOI: 10.1109/JSEN.2019.2909195.

[30] A. Ferrari, D. Micucci, M. Mobilio, and P. Napoletano, "Human Activities Recognition Using Accelerometer and Gyroscope," in *Proceedings of the 15th European Conference on Ambient Intelligence*, series AmI '19, 2019, pages 357–362. DOI: 10.1007/978-3-030-34255-5_28.

[31] P. F. Christ *et al.*, "Diabetes60 – Inferring Bread Units From Food Images Using Fully Convolutional Neural Networks," in *Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops*, series ICCVW '17, 2017, pages 1526–1535. DOI: 10.1109/ICCVW.2017.180.

[32] X. Zhang, F. Zhuang, W. Li, H. Ying, H. Xiong, and S. Lu, "Inferring Mood Instability via Smartphone Sensing: A Multi-View Learning Approach," in *Proceedings of the 27th ACM International Conference on Multimedia*, series MM '19, 2019, pages 1401–1409. DOI: 10.1145/3343031.3350957.

[33] Y. S. Can, B. Arnrich, and C. Ersoy, "Stress detection in daily life scenarios using smart phones and wearable sensors: A survey," *Journal of Biomedical Informatics*, volume 92, 103139:1–103139:22, 2019. DOI: 10.1016/j.jbi.2019.103139.

[34] Z. Mian, K. L. Hermayer, and A. Jenkins, "Continuous Glucose Monitoring: Review of an Innovation in Diabetes Management," *The American Journal of the Medical Sciences*, volume 358, number 5, pages 332–339, 2019. DOI: 10.1016/j.amjms.2019.07.003.

[35] E. Borelli *et al.*, "HABITAT: An IoT Solution for Independent Elderly," *Sensors*, volume 19, number 5, pages 1–23, 2019. DOI: 10.3390/s19051258.

[36] E. Thorstensen, "Privacy and Future Consent in Smart Homes as Assisted Living Technologies," in *Proceedings of the 4th International Conference on Human Aspects of IT for the Aged Population*, series ITAP '18, 2018, pages 415–433. DOI: 10.1007/978-3-319-92037-5_30.

[37] C. Stach, F. Steimle, and A. C. Franco da Silva, "TIROL: The Extensible Interconnectivity Layer for mHealth Applications," in *Proceedings of the 23rd International Conference on Information and Software Technologies*, series ICIST '17, 2017, pages 190–202. DOI: 10.1007/978-3-319-67642-5_16.

[38] C. Stach, F. Steimle, and B. Mitschang, "How to Realize Device Interoperability and Information Security in mHealth Applications," in *Biomedical Engineering Systems and Technologies: 11th International Joint Conference, BIOSTEC 2018, Funchal, Madeira, Portugal, January 19-21, 2018, Revised Selected Papers*, A. Cliquet Jr. *et al.*, Eds. Springer, 2019, pages 213–237. DOI: 10.1007/978-3-030-29196-9_12.

[39] C. Stach and B. Mitschang, "Privacy Management for Mobile Platforms – A Review of Concepts and Approaches," in *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management*, series MDM '13, 2013, pages 305–313. DOI: 10.1109/MDM.2013.45.

[40] ——, "Design and Implementation of the Privacy Management Platform," in *Proceedings of the 2014 IEEE 15th International Conference on Mobile Data Management*, series MDM '14, 2014, pages 69–72. DOI: 10.1109/MDM.2014.14.

[41] A. Sunyaev, "Evaluation of Microsoft HealthVault and Google Health personal health records," *Health and Technology*, volume 3, pages 3–10, 2013. DOI: 10.1007/s12553-013-0049-4.

[42] C. Stach, C. Giebler, M. Wagner, C. Weber, and B. Mitschang, "AMNESIA: A Technical Solution towards GDPR-compliant Machine Learning," in *Proceedings of the 6th International Conference on Information Systems Security and Privacy*, series ICISSP '20, volume 1, 2020, pages 21–32. DOI: 10.5220/0008916700210032.

[43] Y. Ning, Y. Zhu, R.-c. Wand, R. Malekian, and L. Qiao-min, "An Efficient Authentication and Access Control Scheme for Perception Layer of Internet of Things," *Applied Mathematics & Information Sciences*, volume 8, number 4, pages 1617–1624, 2014.

[44] M. Hüffmeyer and U. Schreier, "Analysis of an Access Control System for RESTful Services," in *Proceedings of the 16th International Conference on Web Engineering*, series ICWE '16, 2016, pages 373–380. DOI: 10.1007/978-3-319-38791-8_22.

[45] K. Olejnik, I. Dacosta, J. S. Machado, K. Huguenin, M. E. Khan, and J.-P. Hubaux, "SmarPer: Context-Aware and Automatic Runtime-Permissions for Mobile Devices," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, series SP '17, 2017, pages 1058–1076. DOI: 10.1109/SP.2017.25.

[46] S. Alpers *et al.*, "PRIVACY-AVARE: An approach to manage and distribute privacy settings," in *Proceedings of the 2017 3rd IEEE International Conference on Computer and Communications*, series ICCC '17, 2017, pages 1460–1468. DOI: 10.1109/CompComm.2017.8322784.

[47] G. Cugola and A. Margara, "Processing Flows of Information: From Data Stream to Complex Event Processing," *ACM Computing Surveys*, volume 44, number 3, 15:1–15:62, 2012. DOI: 10.1145/2187671.2187677.

[48] S. M. Palanisamy, F. Dürr, M. A. Tariq, and K. Rothermel, "Preserving Privacy and Quality of Service in Complex Event Processing Through Event Reordering," in *Proceedings of the 12th ACM International Conference on Distributed and Event-based Systems*, series DEBS '18, 2018, pages 40–51. DOI: 10.1145/3210284.3210296.

[49] K. Birman, M. Jelasity, R. Kleinberg, and E. Tremel, "Building a Secure and Privacy-Preserving Smart Grid," *ACM SIGOPS Operating Systems Review*, volume 49, number 1, pages 131–136, 2015. DOI: 10.1145/2723872.2723891.

[50] D. L. Quoc, M. Beck, P. Bhatotia, R. Chen, C. Fetzer, and T. Strufe, "PrivApprox: Privacy-Preserving Stream Analytics," in *Proceedings of the 2017 USENIX Annual Technical Conference*, series USENIX ATC '17, 2017, pages 659–672.

[51] N. Johnson, J. P. Near, and D. Song, "Towards Practical Differential Privacy for SQL Queries," *Proceedings of the VLDB Endowment*, volume 11, number 5, pages 526–539, 2018. DOI: 10.1145/3187009.3177733.

[52] D. Wang, J. Ren, C. Xu, J. Liu, Z. Wang, and X. Zhang Yaoxue Shen, "PrivStream: Enabling Privacy-Preserving Inferences on IoT Data Stream at the Edge," in *Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems*, series HPCC/SmartCity/DSS '19, 2019, pages 1290–1297. DOI: 10.1109/HPCC/SmartCity/DSS.2019.00180.

[53] M. Pourahmadi, "Estimation and Interpolation of Missing Values of a Stationary Time Series," *Journal of Time Series Analysis*, volume 10, number 2, pages 149–169, 1989. DOI: 10.1111/j.1467-9892.1989.tb00021.x.

[54] B. Ramosaj and M. Pauly, "Predicting missing values: A comparative study on non-parametric approaches for imputation," *Computational Statistics*, volume 34, pages 1741–1764, 2019. DOI: 10.1007/s00180-019-00900-3.

[55] T. Sakamoto, M. Yokozawa, H. Toritani, M. Shibayama, N. Ishitsuka, and H. Ohno, "A crop phenology detection method using time-series MODIS data," *Remote Sensing of Environment*, volume 96, number 3, pages 366–374, 2005. DOI: 10.1016/j.rse.2005.03.008.

[56] K. Autio *et al.*, "Food structure and its relation to starch digestibility, and glycaemic response," in *Proceedings of the 3rd International Symposium on Food Rheology and Structure*, series ISFRS '03, 2003, pages 7–11.

[57] L. Billings and I. B. Schwartz, "Exciting chaos with noise: Unexpected dynamics in epidemic outbreaks," *Journal of Mathematical*

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Biology, volume 44, number 1, pages 31–48, 2002. DOI: 10.1007/s002850100110.

[58] KAYAK Germany GmbH. (2020). "PyPika – Python Query Builder," [Online]. Available: https://pypika.readthedocs.io.

[59] C. Stach and B. Mitschang, "The Secure Data Container: An Approach to Harmonize Data Sharing with Information Security," in *Proceedings of the 2016 17th IEEE International Conference on Mobile Data Management*, series MDM '16, 2016, pages 292–297. DOI: 10.1109/MDM.2016.50.

[60] ——, "CURATOR—A Secure Shared Object Store: Design, Implementation, and Evaluation of a Manageable, Secure, and Performant Data Exchange Mechanism for Smart Devices," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, series SAC '18, 2018, pages 533–540. DOI: 10.1145/3167132.3167190.

[61] C. Stach and F. Steimle, "Recommender-based Privacy Requirements Elicitation – EPICUREAN: An Approach to Simplify Privacy Settings in IoT Applications with Respect to the GDPR," in *Proceedings of the 34th Annual ACM Symposium on Applied Computing*, series SAC '19, 2019, pages 1500–1507. DOI: 10.1145/3297280.3297432.

[62] J. Rowley, "The wisdom hierarchy: Representations of the DIKW hierarchy," *Journal of Information Science*, volume 33, number 2, pages 163–180, 2007. DOI: 10.1177/0165551506070706.

[63] C. Stach and B. Mitschang, "ACCESSORS: A Data-Centric Permission Model for the Internet of Things," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, series ICISSP '18, volume 1, 2018, pages 30–40. DOI: 10.5220/0006572100300040.

[64] C. Gritti, M. Önen, and R. Molva, "Device Identification and Personal Data Attestation in Networks," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications*, volume 9, number 4, pages 1–25, 2018. DOI: 10.22667/JOWUA.2018.12.31.001.

[65] C. Stach, F. Steimle, C. Gritti, and B. Mitschang, "PSSST! The Privacy System for Smart Service Platforms: An Enabler for Confidable Smart Environments," in *Proceedings of the 4th International Conference on Internet of Things, Big Data and Security*, series IoTBDS '19, volume 1, 2019, pages 57–68. DOI: 10.5220/0007672900570068.

[66] InfluxData Inc. (2020). "Kapacitor," [Online]. Available: https://www.influxdata.com/time-series-platform/kapacitor/.

[67] C. Giebler, C. Stach, H. Schwarz, and B. Mitschang, "BRAID — A Hybrid Processing Architecture for Big Data," in *Proceedings of the 7th International Conference on Data Science, Technology and Applications*, series DATA '18, volume 1, 2018, pages 294–301. DOI: 10.5220/0006861802940301.

[68] C. Gritti, R. Molva, and M. Önen, "Lightweight Secure Bootstrap and Message Attestation in the Internet of Things," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, series SAC '18, 2018, pages 775–782. DOI: 10.1145/3167132.3167218.

[69] C. Gritti, M. Önen, and R. Molva, "CHARIOT: Cloud-Assisted Access Control for the Internet of Things," in *Proceedings of the 16th Annual Conference on Privacy, Security and Trust*, series PST '18, 2018, pages 1–6. DOI: 10.1109/PST.2018.8514217.

[70] ——, "Privacy-preserving delegatable authentication in the Internet of Things," in *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, series SAC '19, 2019, pages 861–869. DOI: 10.1145/3297280.3297365.

[71] C. Stach, C. Gritti, and B. Mitschang, "Bringing Privacy Control Back to Citizens: DISPEL — A Distributed Privacy Management Platform for the Internet of Things," in *Proceedings of the 35th ACM/SIGAPP Symposium On Applied Computing*, series SAC '20, 2020, pages 1272–1279. DOI: 10.1145/3341105.3375754.

[72] P. Montesano, M. Hueffmeyer, and U. Schreier, "Outsourcing Access Control for a Dynamic Access Configuration of IoT Services," in *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security*, series IoTBDS '17, volume 1, 2017, pages 59–69. DOI: 10.5220/0006257000590069.

[73] C. Stach and B. Mitschang, "Elicitation of Privacy Requirements for the Internet of Things Using ACCESSORS," in *Information Systems Security and Privacy: 4th International Conference, ICISSP 2018, Funchal - Madeira, Portugal, January 22-24, 2018, Revised Selected Papers*, P. Mori, S. Furnell, and O. Camp, Eds. Springer, 2019, pages 40–65. DOI: 10.1007/978-3-030-25109-3_3.

[74] M. Rhif, A. B. Abbes, I. R. Farah, B. Martínez, and Y. Sang, "Wavelet Transform Application for/in Non-Stationary Time-Series Analysis: A Review," *Applied Sciences*, volume 9, number 7, 1345:1–1345:22, 2019. DOI: 10.3390/app9071345.

[75] X. Li, R. Shen, and R. Chen, "Improving Time Series Reconstruction by Fixing Invalid Values and its Fidelity Evaluation," *IEEE Access*, volume 8, pages 7558–7572, 2020. DOI: 10.1109/ACCESS.2019.2962757.

[76] Y.-S. Moon, H.-S. Kim, S.-P. Kim, and E. Bertino, "Publishing Time-Series Data under Preservation of Privacy and Distance Orders," in *Proceedings of the 21st International Conference on Database and Expert Systems Applications*, series DEXA '10, 2010, pages 17–31. DOI: 10.1007/978-3-642-15251-1_2.

[77] M.-J. Choi, H.-S. Kim, and Y.-S. Moon, "Publishing Sensitive Time-Series Data under Preservation of Privacy and Distance Orders," *International Journal of Innovative Computing, Information and Control*, volume 8, number 5(B), pages 3619–3638, 2012.

[78] G. R. Lee, R. Gommers, F. Wasilewski, K. Wohlfahrt, and A. O'Leary, "PyWavelets: A Python package for wavelet analysis," *Journal of Open Source Software*, volume 4, number 36, 1237:1–1237:2, 2019. DOI: 10.21105/joss.01237.

[79] ——, (2020). "PyWavelets – Wavelet Transforms in Python," [Online]. Available: https://pywavelets.readthedocs.io.

[80] A. Gorelik, *The Enterprise Big Data Lake: Delivering the Promise of Big Data and Data Science*, A. Oram, Ed. Sebastopol, CA: O'Reilly Media, Inc., 2019, ISBN: 978-1-491-93155-4.

[81] C. Giebler, C. Gröger, E. Hoos, H. Schwarz, and B. Mitschang, "Leveraging the Data Lake: Current State and Challenges," in *Proceedings of the 21st International Conference on Big Data Analytics and Knowledge Discovery*, series DaWaK '19, 2019, pages 179–188. DOI: 10.1007/978-3-030-27520-4_13.

[82] ——, "A Zone Reference Model for Enterprise-Grade Data Lake Management," in *Proceedings of the 24th IEEE Enterprise Computing Conference*, series EDOC '20, 2020, pages 1–10.

[83] B. Sharma, *Architecting Data Lakes: Data Management Architectures for Advanced Business Use Cases*, 2nd Edition, R. Roumeliotis, Ed. Bejing *et al.*: O'Reilly Media, Inc., 2018, ISBN: 978-1-492-03299-1.

[84] R. Korte, J. Bräcker, and J. Brockmeyer, "Gastrointestinal digestion of hazelnut allergens on molecular level: Elucidation of degradation kinetics and resistant immunoactive peptides using mass spectrometry," *Molecular Nutrition & Food Research*, volume 61, number 10, 1700130:1–1700130:16, 2017. DOI: 10.1002/mnfr.201700130.

[85] C. Stach, C. Gritti, D. Przytarski, and B. Mitschang, "Trustworthy, Secure, and Privacy-aware Food Monitoring Enabled by Blockchains and the IoT," in *Proceedings of the 2020 IEEE International Conference on Pervasive Computing and Communications Workshops*, series PerCom '20, 2020, pages 1–4. DOI: 10.1109/PerComWorkshops48775.2020.9156150.

[86] R. Eichler, C. Giebler, C. Gröger, H. Schwarz, and B. Mitschang, "HANDLE – A Generic Metadata Model for Data Lakes," in *Proceedings of the 22nd International Conference on Big Data Analytics and Knowledge Discovery*, series DaWaK '20, 2020, pages 1–15.

[87] I. Litou, V. Kalogerakia, I. Katakis, and D. Gunopulos, "Real-Time and Cost-Effective Limitation of Misinformation Propagation," in *Proceedings of the 2016 17th IEEE International Conference on Mobile Data Management*, series MDM '16, 2016, pages 158–163. DOI: 10.1109/MDM.2016.33.

[88] ——, "Efficient and timely misinformation blocking under varying cost constraints," *Online Social Networks and Media*, volume 2, pages 19–31, 2017. DOI: 10.1016/j.osnem.2017.07.001.

[89] D. Przytarski, C. Stach, C. Gritti, and B. Mitschang, "A Blueprint for a Trustworthy Health Data Platform Encompassing IoT and Blockchain Technologies," in *Proceedings of the 29th International Conference on Software Engineering and Data Engineering*, series SEDE '20, 2020, pages 1–10.

[90] D. Przytarski, "Using Triples as the Data Model for Blockchain Systems," in *Proceedings of the 18th International Semantic Web Conference*, series BlockSW/CKG@ISWC '19, 2019, 4:1–4:2.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Reducing the Attack Surface for Sensitive Data

George O. M. Yee
Computer Research Lab, Aptusinnova Inc., Ottawa, Canada
Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
e-mail: george@aptusinnova.com, gmyee@sce.carleton.ca

Abstract—**Breaches of sensitive data have been occurring at an alarming rate to the embarrassment and expense of companies. It would appear that in each breach, the attack surface for the data has been sufficiently large to attract attackers. Reducing this attack surface is a way to lessen the likelihood of breaches. This paper presents methods for reducing the attack surface of the data held in the online computer systems of organizations. The methods are applied to a software system's architecture early in the design process, as an approach for designing-in security. This work first defines the attack surface and then uses this definition to obtain methods for reducing the attack surface. The definition also leads to a formula for calculating the size of the attack surface. The formula incorporates the fact that vulnerabilities differ within the architecture. This paper further gives recommendations on how to apply the methods effectively and illustrates this application using two examples. Reducing the attack surface may not prevent breaches, but it will make them less likely to occur.**

Keywords-sensitive data; private data; breaches; attack surface identification; attack surface reduction.

I. INTRODUCTION

This work extends Yee [1] by a) extending the application domain to all sensitive data, not just private data, b) improving the calculation of the size of the attack surface to account for the fact that some parts of the software architecture are more likely to be attacked than others, c) improving the explanations throughout the paper as well as updating the examples of breaches in Section I, d) adding a second application example, and e) increasing the number of references.

Breaches of sensitive data held by companies and other types of organizations have been occurring at an alarming rate. In recent years, each year has been accompanied by its assortment of data breaches. Consider the following sampling of breaches in 2020 [2], the year of this work:

- July 20, 2020: Ancestry.com, an unsecured server exposed sensitive data belonging to 60,000 clients of this family history search company. The lost data include email addresses, geolocation information, IP addresses, system user IDs, support messages and technical support details.
- June 22, 2020: BlueLeaks, over 296 GB of data was leaked from US law enforcement agencies and fusion centers, and posted on an online searchable portal called BlueLeaks. The leaked information contained more than one million files, including scanned documents, videos, emails, audio files, some of which had sensitive and personal data, such as names, bank account numbers, and phone numbers.
- April 20, 2020: Beaumont Health, the personal and medical data of more than 112,000 employees and patients of Beaumont Health was accessed by a hacker after compromising employee email accounts using a phishing attack. The lost data included names, birth dates, social security numbers, driver's license numbers, medical condition information, and bank account data.
- February 20, 2020: MGM Resorts, the personal information of over 10.6 million hotel guests who had stayed at MGM Resorts was found posted on a hacking forum. The information included names, home addresses, phone numbers, emails, and dates of birth. On July 15, 2020, researchers found 142 million personal records of formers guests of MGM Resorts hotels for sale on the Dark Web, suggesting that the original breach was larger than previously announced.

Apparently, the attack surface for the data that was breached, or the number of ways that the data could be accessed and stolen, was sufficiently large and attractive to the attackers.

Given the rate of recent data breaches, it is clear that more needs to be done to reduce the probability of a data breach occurring. The objective of this work is to derive methods for reducing the attack surface of sensitive data held in online (i.e., connected to the Internet) computer systems of organizations. The methods are obtained from consideration of the definition of the attack surface, which in turn is based on how an attack happens. This definition also leads to a straightforward formula for calculating the size of the attack surface, which can be used to verify that use of the methods does indeed reduce the attack surface. The methods focus on reducing the attack surface by altering the system architecture, rather than the deployment of add-on security appliances, such as firewalls and intrusion detection systems. The methods are meant to be applied at the early stages of design within a software development cycle, as part of the Design for Security toolset.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

This paper is organized as follows. Section II explains sensitive data, attacks, and attack surface. Section III derives methods for reducing the attack surface. Section IV illustrates the methods using two application examples. Section V describes related work. Section VI discusses some potential issues. Finally, Section VII presents conclusions and future work.

II. SENSITIVE DATA, ATTACKS, AND ATTACK SURFACE

This section explains sensitive data, attacks, attack surface, and how to calculate the size of the attack surface.

A. Sensitive Data, Attacks, and Attack Surface

Sensitive data is data that needs protection and must not fall into the wrong hands. It includes private or personal data. Sensitive data also includes non-private information that may compromise the competitiveness of the organization if divulged, such as trade secrets or proprietary algorithms and formulas. For government organizations, non-personal sensitive data may include information that is vital for the security of the country for which the government organization is responsible.

Private data, also known as personal data, is data about an individual, can identify that individual, and is owned by that individual [3]. For example, an individual's driver license number, passport number, or credit card number can each be used to identify the individual and are therefore considered as private data. The individual's privacy then refers to his/her ability to control the collection (what personal data and collected by which party), purpose of collection, retention, and disclosure of that data, as stated in the individual's privacy preferences [3].

DEFINITION 1: *Sensitive data* (SD) is information that must be protected from unauthorized access in order to safeguard the privacy of an individual, the well-being or expected operation of an organization, or the well-being or expected functioning of an entity for which the organization has responsibility.

DEFINITION 2: An *attack* is any action carried out against an organization's computer system that, if successful, results in the system being compromised.

This work focuses on attacks that compromise the SD held in the online systems of organizations. The attacker who launches an attack may be internal (inside attacker) or external (outside attacker) to the organization. This work applies to both types of attackers. An internal attacker usually has easier access to the targets of his/her attack and he/she may hide his/her attacks in the guise of normal duty.

Salter et al. [4] give an interesting insight into what enables a successful attack: "Any successful attack has three steps: One, diagnose the system to identify some attack. Two, gain the necessary access. And three, execute the attack. To protect a system, only one of these three steps needs to be blocked." Thus, an attack surface must contain a target that the attacker deems worthy of attack (suit his/her purpose for the attack) and that target must be accessible to the attacker. For this work, the target that is potentially worthy of attack is the SD that is accessible to attackers. In a computer system, this SD is either moving (travelling from one location to another), at rest (stored), or being used (by some process). This leads to the following definition of attack surface:

DEFINITION 3: The *attack surface* for sensitive data, also called the *data attack surface*, contained in an online computer system is the set of all locations in the system that contain attacker accessible SD in the clear, where the SD is moving, at rest, or being processed.

In Definition 3, "attacker accessible SD" means that the attacker is able to exfiltrate the SD using some agent of attack, such as malware against stored SD and SD being processed, or a man-in-the-middle attack against a link containing moving SD. Also, we assume that attackers would attack SD that is in the clear rather than SD that is encrypted. In the rest of this paper, by "attack surface" we mean the data attack surface, unless otherwise indicated. Figure 1 shows an example data attack surface.

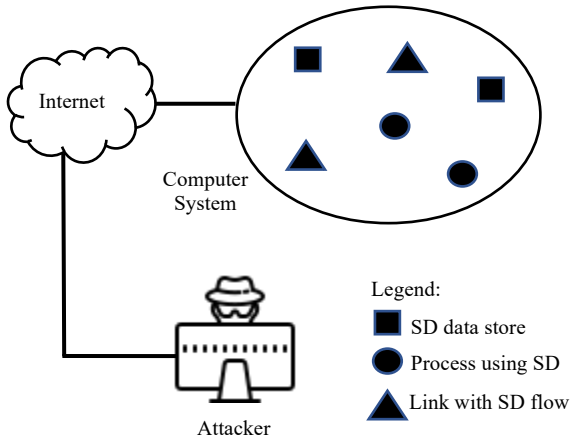


Figure 1. Example data attack surface consisting of the set of all 6 attacker accessible locations in the system that contain SD in the clear.

An alternative definition of attack surface for SD contained in a computer system is the set of ways the attacker has to exfiltrate the SD. However, given the complexity of computer systems and the fact that the tools available to the attacker to use in his/her attacks are unknown to us, it is next to impossible to determine this set. On the other hand, locations that contain attacker accessible SD are easier to identify. Since an exfiltration must be from a location that contains SD, the set of such exfiltrations depends on the set of such locations. The larger the set of locations, the larger the set of exfiltrations. The smaller the set of locations, the smaller the set of exfiltrations. Therefore, Definition 3 in a sense includes this alternative definition, but in addition, is more easily applied.

As mentioned above, in the first step of a successful attack, the attacker diagnoses the system to identify the attack [4]. A smaller attack surface will make this step more difficult for the attacker. Therefore, a smaller attack surface corresponds to higher security, which is why we wish to reduce the attack surface. Definition 3 also gives rise to this conclusion: a smaller attack surface means a smaller number

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

of locations that contain SD, which in turn means fewer opportunities for exfiltration of the SD, or in other words, higher security.

Definition 3 is consistent with the intuitive understanding of an attack surface (the usual meaning), which is "the set of ways in which an adversary can enter the system and potentially cause damage" [5]. Each "way" corresponds to a location in Definition 3 that in turn corresponds to methods for exfiltrating SD from the location.

B. Calculating the Size of the Attack Surface

It would be useful to have a numerical value for the size of the attack surface, since then we could a) compare attack surfaces at different stages of development to see if the system's security is getting better or worse, b) compare attack surfaces of different systems when choosing a system for purchase, and c) easily see if actions taken to reduce the attack surface have indeed reduced it.

As mentioned above, sensitive data held in a computer system can be in the following three states: moving, at rest, or being processed. These states correspond respectively, within a computer system, to SD that is moving along a link, SD that is stored in a data store, and SD that is being processed. Thus, the locations in Definition 3 refer to links, datastores, and processes that contain attacker accessible SD. Definition 3 then leads naturally to the following formula for calculating the size of the attack surface for sensitive data.

Let N be an *estimate* of the size of the attack surface for SD. Let m, n, and k be the number of links, data stores, and processes, respectively, that contain attacker accessible SD in the clear. Then

$$N = m + n + k \tag{1}$$

Equation (1) says that N is found by adding up the number of attacker accessible locations in the system that contain SD, namely: the number m of links, the number n of data stores, and the number k of processes, all of which contain attacker accessible SD. This equation follows directly from Definition 3, by simply replacing "attack surface" with "size of the attack surface" and "set" with "size of the set" in that definition. We call N an estimate because it is impossible to know all the vulnerabilities in a system, and hence it is impossible to have an accurate value of the size of the attack surface. As well, an estimate suffices for the three benefits mentioned at the beginning of this Section. In the following, unless stated otherwise, we use "size" to mean "estimated size".

Equation (1) is the minimum size of the attack surface because it counts the number of links, data stores, and processes that form the attack surface. It can only be smaller if one or more of these components are not part of the attack surface, which would contradict Definition 3. Further, (1) makes no distinction between links, data stores, or processes as locations that contain attacker accessible SD. Yet, given a choice between attacking a link, a data store, or a process in the same computer system, where the difficulty level is the same for all locations, the attacker will probably choose to attack a data store of SD as it is more likely to give the attacker what he/she wants. This preference for data stores is seen in the large number of data store breaches that have occurred. Thus, a data store should contribute more to the size of the attack surface than a link or a process. A datastore makes the system more vulnerable to attack and this should be reflected in the size of the attack surface, which is also a measure of the system's vulnerability to attack. We can reflect this in (1) by making the contribution of data stores to the size of the attack surface as $M_d \bullet n$ where M_d is a positive integer multiplier. In fact, we can have positive integer multipliers M_l for links and M_p for processes, and corresponding contributions $M_l \bullet m$ and $M_p \bullet k$ to the size of the attack surface. Getting back to reflecting the greater contribution of data stores to the size, we can set $M_l = M_p = 1$ and $M_d = 2$, which states that each data store contributes twice as much to the size of the attack surface as either a link or a process. Thus, the equation for the size of the attack surface with multipliers is

$$N = M_l m + M_d n + M_p k \tag{2}$$

and with the above values of the multipliers to reflect the increased contribution to size of data stores, (2) becomes

$$N = m + 2n + k \tag{3}$$

We will use (3) rather than (1), since it suits our goal and is closer to reality. Applying (1) to Figure 1 gives an attack surface of size $N = m + n + k = 2 + 2 + 2 = 6$. Applying (3) gives 8, reflecting the greater vulnerability of the system due to its data stores.

We have used $M_l = M_p = 1$ and $M_d = 2$ with the rationale that attackers are more attracted to data stores than to links and processes, plus the observation that there has been many breaches of data stores. Note that $M_l = M_p = 2$ and $M_d = 3$ would have worked as well. The effect of these values in reducing the attack surface is that eliminating a data store from the attack surface gives a greater reduction than removing a link or a process. This means that if a data store can be removed, it should be. However, the values can be anything so long as they relatively reflect what the contributions to size should be according to past history or other sources of information, such as the system architecture. Since we don't have access to other values, we have used the ones above since they reflect our conviction regarding datastores. Perhaps in the future, these values can be refined based on studies. Note that whatever values are used for M_l, M_d, and M_p, our attack surface reduction techniques will always show N decreasing after applying each technique. Note also that (2) treats all links equally, all data stores equally, and all processes equally in terms of their contributions to the size of the attack surface. To allow each location to have its own specific contribution would be to attempt a level of accuracy that is unwarranted in light of our lack of knowledge of attacker behaviour as well as all the vulnerabilities in the system.

III. Reducing the Attack Surface

This section derives methods for reducing the attack surface based on (3).

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

A. Methods for Reducing the Attack Surface

Equation (3) implies that the attack surface will decrease if and only if any or all of the quantities m, n, or k decrease. Therefore, the attack surface may be reduced by the following methods, where each method decreases m, n, or k.

a) Make a SD location useless to the attacker.
b) Combine two or more SD locations into a single SD location.
c) Deny the attacker access to a SD location.
d) Remove a SD location from the system.

The following explains these methods in greater detail and describes how they may be carried out.

a) *Make* a *SD Location Useless to the Attacker*

As mentioned above, in the first step of a successful attack, the attacker diagnoses the system to identify an attack, or in our case, the SD target for the attack. In this diagnosis, it is reasonable to assume that the attacker will ignore any target that he/she finds useless for his/her purposes. Such targets may be removed from the attack surface. Some ways to make a SD target useless to an attacker are:

- Obfuscate (e.g., encrypt) the SD at the location. The attacker will not want to exfiltrate SD that cannot be read. The computer system will need to be able to de-obfuscate the data securely for its own purposes.
- Anonymize the SD at the location. Again, the attacker will not want SD that cannot be linked to individuals, since it is this linking that adds value to the data, e.g., for advertising purposes. The computer system will need to be able to de-anonymize the data securely for its own purposes.

Note that this method does not affect any other location, whereas the following methods do.

To illustrate, obfuscating one data store and one process in Figure 1 results in Figure 2, where the obfuscated data store and the obfuscated process have been removed from the attack surface. It can be seen that the attack surface in Figure 2 is reduced (size 5) relative to the attack surface of Figure 1 (size 8).

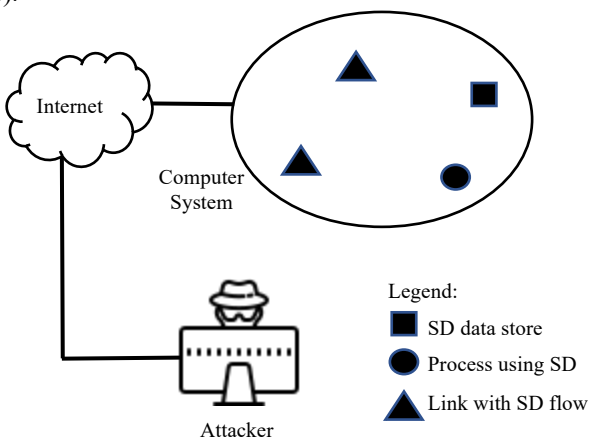


Figure 2. Resulting reduced data attack surface of size 5 after obfuscating locations in Figure 1.

b) *Combine Two or More SD Locations into a Single SD Location*

This method will decrease the number of SD locations and reduce the size of the attack surface per (3). Links carrying SD to/from the locations that were combined may need to be moved to the combined location, or they may be merged if they extend from the locations that were combined to a common endpoint. Merged SD carrying links correspond to a reduction of SD carrying links from the attacks surface. In addition, changes to the software logic may be needed for data stores or processes that were combined to accomplish reading or storing the data in the combined location (for combined data stores), or new processing of data in the combined location (for combined processes). As we have seen above, this method can remove SD carrying links from the attack surface when such links can be merged.

To illustrate, suppose in Figure 1 that we combine two data stores and two processes into one data store and one process. Suppose also that combining the data stores allowed the merging of two links into one, but combining the processes did not change the number of links. Figure 3 shows the result, obtained by removing 1 data store, 1 process, and 1 link from Figure 1 due to combining locations. We see that the attack surface has been reduced from size 8 (Figure 1) to size 4 (Figure 3).

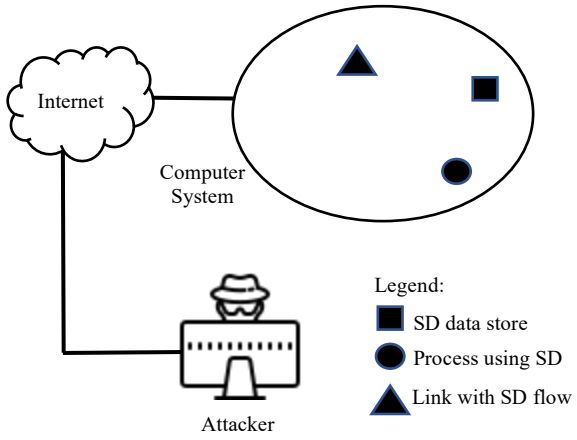


Figure 3. Resulting reduced data attack surface of size 4 after combining (method b) or removing (method d) locations in Figure 1.

c) *Deny the Attacker Access to a SD Location*

It may be possible to have some SD locations offline, thus denying the attacker access to these locations. For example, this may be possible for certain self-contained processing, such as analytics, that can be done using SD that is offline. In this case, all data stores, processes, and data links involved solely in the processing to be moved offline may be removed from the attack surface of the system and re-constituted into the offline system. It may be necessary to update the offline SD data stores periodically using data from the system that is online. This update will need to be done in a secure fashion, perhaps by transferring the data manually using disks, after making sure that no malware can infect the offline system via this transfer. Although the destination locations are offline, it

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

may still be possible for transferred malware to exfiltrate the offline data, e.g., hiding the data in the disks that are used for transfer and then transmitting the data once the disks are on the online part of the system. The locations moved offline are still vulnerable to inside attack, so they would have to be secured against such attack. Defending against inside attacks has been extensively researched, e.g., [6], [7].

Another way to deny the attacker access to a SD location applies to SD links. Here, SD links are implemented on a hardware platform along with other components of the system. An example of such a link is the communication channel between the CPU and the GPU implemented on a computer motherboard. An attacker would find it very difficult to access such links for an attack such as man-in-the-middle. Given other targets that are easier to access, the attacker will not attack such links and they can be removed from the attack surface.

As an illustration of moving some SD locations offline, Figure 4 shows a data store, a process, and a link taken out of Figure 1 and assembled into an offline system. The attack surface of the computer system has been reduced from size 8 (Figure 1) to size 4 (subtracting the contributions to the attack surface of the moved locations). However, the offline system would need to be secured against inside attack.

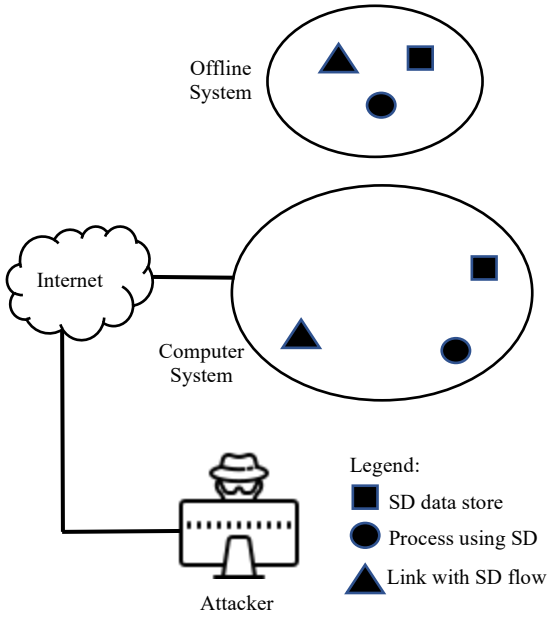


Figure 4. Resulting reduced data attack surface of size 4 for the computer system after moving some locations in Figure 1 offline.

d) Remove a SD Location from the System

Another way to reduce the attack surface is to remove a SD location from the system by deciding that the SD in the location is no longer required. For example, a company that stores the credit card information of its customers for their convenience may decide to stop storing this information, and instead, ask the customer for their credit card information every time the customer goes through checkout. This is in general a good decision, to avoid storing SD that may get compromised, at the cost of a little inconvenience. In this case, the associated credit card SD datastore would no longer be needed, and would be removed from the attack surface. Another example is the removal of a process that periodically sends customers the status of their order. The process uses SD consisting of the customer's name and email address to send the status. Suppose that this process is no longer necessary because the customer can now use a new Web interface to check order status. Removal of this process from the system removes it from the attack surface. Interestingly, removal of a SD location can also result in removing other SD locations that are connected to the location that is removed. For example, the removal of a SD data store or a process that uses SD can result in also removing connected SD locations, such as the links that carry SD, or a SD data store that the removed process was exclusively using. Thus, removing a SD location not only removes that location from the attack surface but can also lead to removing other SD locations further reducing the attack surface.

To illustrate, suppose it is decided that one of the processes in Figure 1 is no longer needed. Removing this process means that a data store and a link that were used only by this process are also no longer needed. Thus, the attack surface of the computer system in Figure 1 is reduced from size 8 to size 4, and the reduced system is shown in Figure 3.

B. Applying the Methods

Since the above methods operate on attacker accessible SD locations, it is recommended that they be applied in the second phase of two phases, where the attacker accessible SD locations are identified in the first phase. These phases are carried out on an architectural representation of the online system, such as a Data Flow Diagram (DFD) [8] (see the application examples in Section IV). The phases are as follows.

- Phase 1: Identify SD locations by tracing the flow of sensitive data in the online computer system, looking for where SD enters the system, where SD flows (links), where it is stored (data stores), and where it is used (processes). Identifying the SD locations by tracing the flow of SD in the system implies that there are paths to the SD that an attacker can use to exfiltrate the SD. We therefore conclude that all SD locations found in this manner in an online system are attacker accessible SD locations. Given the ingenuity of attackers (the exfiltration could even be aided by an insider of the organization that owns the computer system, through social engineering), this conclusion is valid.
- Phase 2: Apply the above methods to the attacker accessible SD locations found in Phase 1, where possible, while considering the potential negative effects on the following aspects of the system:
  - Performance
  - Reliability and dependability
  - Ease of maintenance
  - Implementation cost

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

For example, encryption or anonymization incurs extra overhead, combining data stores may introduce a performance bottleneck since the newly combined data store will now need to additionally support data accesses that were originally shared among the data stores that were combined. Combining SD locations in general may reduce modularity and lead to extra effort needed to maintain the system.

Three guiding rules for applying the methods are:

1. Look for opportunities to apply a method to a data store since according to (3), removing a data store from the attack surface gives a greater reduction than removing either a link or a process.
2. Look for opportunities to apply the methods where the potential negative effects mentioned above are minimal.
3. It may be more efficient to consider method a) last, since the other methods can add/delete links that are candidates for method a).

Carrying out the above phases clearly requires knowledge of the computer system in terms of identifying the SD locations. Some basic knowledge of security would also be advantageous. These skills should be found within the software development team responsible for developing the system, perhaps with a little security training if needed.

IV. Application Examples

This section illustrates how to apply the methods for reducing the attack surface for sensitive data using two example online computer systems: one for selling merchandise and the other for airline reservation.

A. Online Seller of Merchandise

Suppose that the system of the online seller of merchandise (e.g., Amazon.com) is at the beginning stages of development and that the development team has produced a DFD showing how sensitive data will flow, be stored, and used in the system. This DFD is shown in Figure 5.

The system in Figure 5 allows the customer to enter his/her "name", "address", "email", "item selected" for purchase, and "credit card info" for payment. These comprise the SD for this example. Five processes cooperate to provide the functionality for the system. One datastore stores the customers' sensitive data; another datastore contains inventory data, i.e., what items are in stock. The system is an online system since it is for an online seller. The SD locations will be found by tracing the flow of SD in the system (described below). All SD locations in the system are attacker accessible SD locations, as noted above in the description of Phase 1 (Section III-B).

Applying Phase 1 in Section III-B, we trace the flow of sensitive data from the point where the data enters the system at process 1. From there, the SD passes through process 1 and is stored in the customer datastore. After this datastore, the SD is split up with the "credit card info" going to process 4 to be used, and the "name", "address", "email", and "item selected" going to process 2, where the "item selected" datum is used, and "name" and "address" are passed to process 5 to print the shipping label, whereas "email" is passed to process 3 to send the customer the shipping status. Thus, we can identify the SD locations as links, datastores, and processes through which the SD passes, is stored, and used. Note that inventory data is not considered SD in this example. These attacker accessible SD locations are shown in Table I.

Table I shows that there are 6 attacker accessible SD link locations, 1 attacker accessible SD datastore, and 5 attacker accessible SD processes. For Figure 5, prior to the application of the above methods, (3) gives the size N of the attack surface for sensitive data as $N = m + 2n + k = 6 + 2 + 5 = 13$.

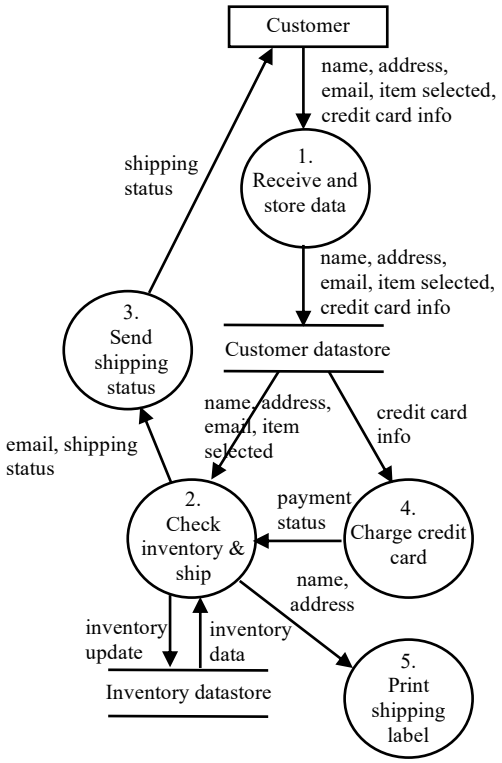


Figure 5. DFD for online seller system, showing how data flows, are stored, and used.

TABLE I. Attacker accessible SD locations in Figure 5

	Links	Datastores	Processes
1	link into process 1	customer datastore	process 1
2	link out of process 1		process 2
3	link from customer datastore to process 2		process 3
4	link from customer datastore to process 4		process 4
5	link into process 5		process 5
6	link into process 3		

Applying Phase 2 in Section III-B, we first use the above methods on the attacker accessible locations in Table I, as follows:

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

- Using method d), remove the customer datastore from the system; it was decided that storing customer SD was not needed (customer purchase history can be stored securely on the customer's device by the seller's website and later retrieved by that same website). Note that removing this data store also caused the removal of a SD link (the link between process 1 and this data store). This change was seen as acceptable, and not significantly impacting performance or system maintainability.
- Using method b), combine process 3 with process 2; this was seen to have negligible impact on performance and an acceptable reduction in modularity.
- Using method b), combine process 5 with process 2; this was also seen to have negligible impact on performance and an acceptable reduction in modularity.

These changes result in the DFD shown in Figure 6. Table II gives the attacker accessible SD locations corresponding to Figure 6.

Table II shows that there are 3 attacker accessible SD link locations and 3 attacker accessible SD processes. For Figure 6, (3) gives the size N of the attack surface for sensitive data as $N = m + 2n + k = 3 + 0 + 3 = 6$. Thus, the application of methods d) and b) have reduced the attack surface from 13 to 6.

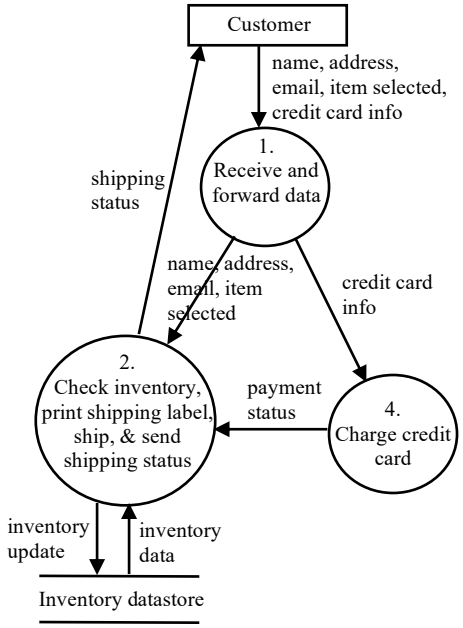


Figure 6. DFD for online seller system after combining processes and removing the customer data store.

We can further reduce the attack surface as follows:

- Using method a), obfuscate (encrypt) the links in Table II; the impact on performance due to the extra over head is deemed acceptable.
- Using method a), obfuscate (encrypt) the SD in the processes shown in Table II; here, the impact on performance and the cost involved for extra code to handle encryption/decryption were considered unacceptable, and this reduction method was not applied. This reduction may have been feasible if these processes could use encrypted SD, but these processes require SD in the clear.

TABLE II. ATTACKER ACCESSIBLE SD LOCATIONS IN FIGURE 6

	Links	Datastores	Processes
1	link into process 1		process 1
2	link from process 1 to process 2		process 2
3	link from process 1 to process 4		process 4

Table III shows the remaining attacker accessible SD locations after applying method a) to the links in Table II. The new attack surface is of size $N = m + 2n + k = 0 + 0 + 3 = 3$. The application of the methods in Section III-A has improved the security of sensitive data in the system by reducing the size of the attack surface from 13 to 3.

TABLE III. REMAINING ATTACKER ACCESSIBLE SD LOCATIONS IN FIGURE 6 AFTER OBFUSCATING THE LINKS IN TABLE II

	Links	Datastores	Processes
1			process 1
2			process 2
3			process 4

Comparing Figure 6 to Figure 5, reducing the attack surface required the following architectural changes to the system: i) eliminating the customer database, ii) reducing the number of processes from 5 to 3 by eliminating processes 3 and 5, and iii) changing the functionality of processes 1 and 2. As noted above, the implications of these changes were accepted by the development team.

The size of the attack surface obtained by applying the above methods may depend on which methods were applied and the order in which they were applied. In particular, it may depend on the available opportunities for applying methods d) and b). For example, by using only method a) (obfuscation) on the link and datastore locations in Table I, assuming that it is not advisable to apply method a) to the processes due to unacceptable impacts on performance and costs, we obtain an attack surface of size 5 (for the remaining 5 processes since the obfuscated links and datastore would have been removed from the attack surface), which is larger than the attack surface of size 3 obtained above by opportunistically applying methods d) and b) before method a). This is the rationale behind guiding rule 3 in the description of Phase 2 above, that it may be more efficient to apply method a) last.

B. Online Airline Reservation System

ACCURES is an online airline reservation system that has been awarded to a software company for development. The system is to consist of 3 modules: MAIN, MOD-CAN, and MOD-EU. MAIN and MOD-CAN operate in Canada,

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

whereas MOD-EU operates in Germany. MAIN communicates with MOD-CAN and MOD-EU to receive flight request details, and in turn, sends them flights assigned details corresponding to the requests. MOD-CAN and MOD-EU process customer transactions, where each transaction consists of receiving customer identification details, flight request details, and payment information (e.g., credit card details), storing the customer information in a data store, processing the payment, and sending the customer his/her travel itinerary, which marks the end of the transaction. The development team creates the DFD for ACCURES shown in Figure 7. Note that in this DFD, all data are SD, and all locations are attacker accessible SD locations.

Applying Phase 1 in Section III-B, we trace the flow of sensitive data from the point where sensitive customer data enters the system at processes 4 and 7. Sensitive data also enters the system in the form of "flight availability updates" but these updates are stored in the Flights data store and go no farther. Referring to Figure 7, we see that the customer SD flows though all locations of MOD-CAN and MOD-EU. We also see that the sensitive data items "flight details requested" and "flight details assigned" flow through all locations of MAIN. Thus, we can conclude that all locations in ACCURES are attacker accessible SD locations, as shown in Table IV.

TABLE IV. ATTACKER ACCESSIBLE SD LOCATIONS IN FIGURE 7

	Links	Datastores	Processes
1	link into Flights data store	Flights	process 1
2	link between Flights data store and process 1	Customer data 1	process 2
3	link between process 1 and process 2	Customer data 2	process 3
4	link between process 2 and process 3		process 4
5	link between process 2 and process 6		process 5
6	link between Customer and process 4		process 6
7	link between process 4 and Customer data 1		process 7
8	link between process 3 and process 4		process 8
9	link between process 3 and Customer data 1		
10	link between Customer data 1 and process 5		
11	link between Customer and process 7		
12	link between process 7 and Customer data 2		
13	link between process 6 and process 7		
14	link between process 6 and Customer data 2		
15	link between Customer data 2 and process 8		

Table IV shows that there are 15 attacker accessible SD link locations, 3 attacker accessible SD datastores, and 8 attacker accessible SD processes. For Figure 7, prior to the application of the above methods, (3) gives the size N of the attack surface for sensitive data as $N = m + 2n + k = 15 + 6 + 8 = 29$.

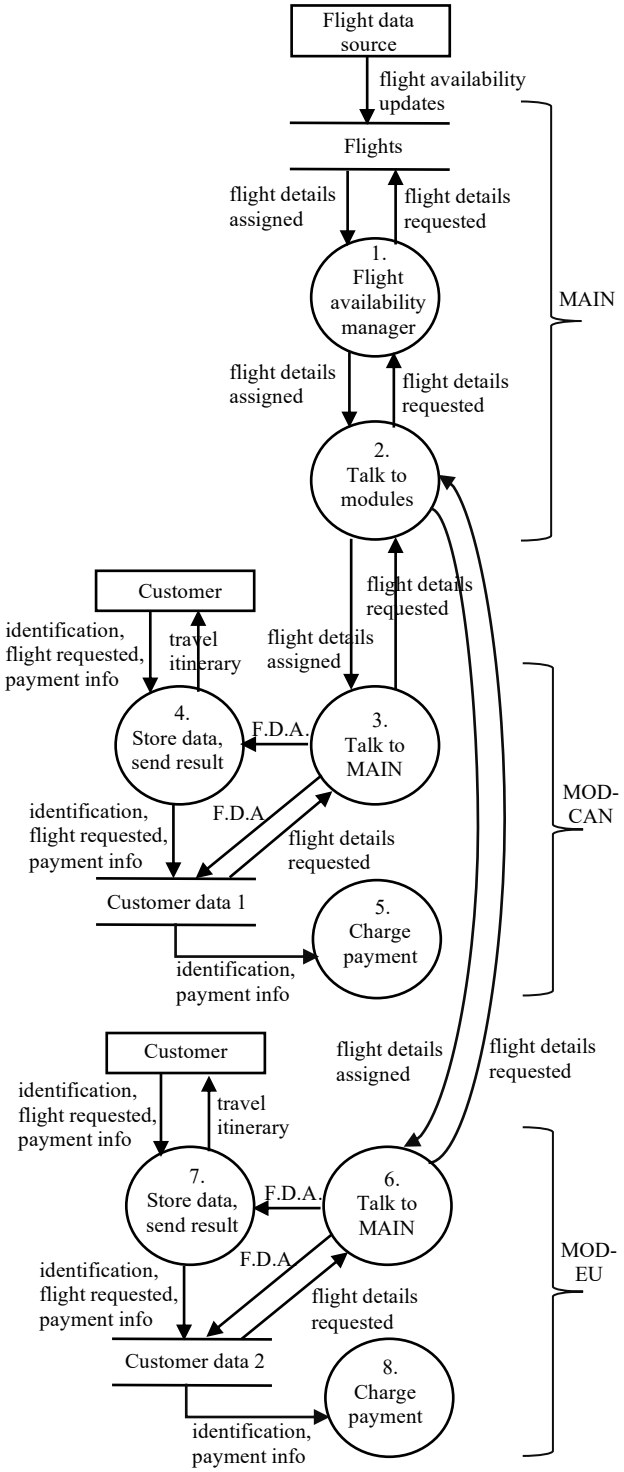


Figure 7. DFD for ACCURES, "F.D.A." stands for "Flight details assigned" – used due to lack of space. All data are SD.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Note that we count a 2-way communication (two arrows in opposite directions) as one link. An example is the 2-way communication between processes 1 and 2. This is because such a communication is considered implemented on a single physical cable, so it is really a single link that will be vulnerable to attack.

Applying Phase 2 in Section III-B, we first use methods b) and d) on the attacker accessible locations in Table IV, as follows:

- Using method b), combine process 5 with process 4 so that process 4 will now take on the additional function of charging the payment. Similarly, combine process 8 with process 7 so that process 7 additionally charges the payment. Note that these applications of method b) also eliminate the link from data store "Customer data 1" to process 5 and the link from data store "Customer data 2" to process 8. These changes were not seen as impacting performance, reliability, or maintenance and were accepted.
- Using method d), remove the two datastores "Customer data 1" and "Customer data 2" from the system; the software company's client decided that storing the customers' sensitive private information in these datastores results in excessive risks that the data could be compromised by attackers. The system would still be able to function according to expectation without these data stores. Note that removing these datastores also caused the removal of 4 SD links that were connecting to the data stores (two links per data store). These changes were also considered feasible and accepted.

These changes result in the DFD shown in Figure 8. Table V gives the attacker accessible SD locations corresponding to Figure 8.

TABLE V. ATTACKER ACCESSIBLE SD LOCATIONS IN FIGURE 8

	Links	Datastores	Processes
1	link into Flights data store	Flights	process 1
2	link between Flights data store and process 1		process 2
3	link between process 1 and process 2		process 3
4	link between process 2 and process 3		process 4
5	link between process 2 and process 6		process 6
6	link between Customer and process 4		process 7
7	link between process 3 and process 4		
8	link between Customer and process 7		
9	link between process 6 and process 7		

Table V shows that there are 9 attacker accessible SD link locations, 1 attacker accessible SD datastore, and 6 attacker accessible SD processes. For Figure 8, (3) gives the size N of the attack surface for sensitive data as $N = m + 2n + k = 9 + 2 + 6 = 17$. Thus, the application of methods b) and d) have reduced the attack surface from 29 to 17.

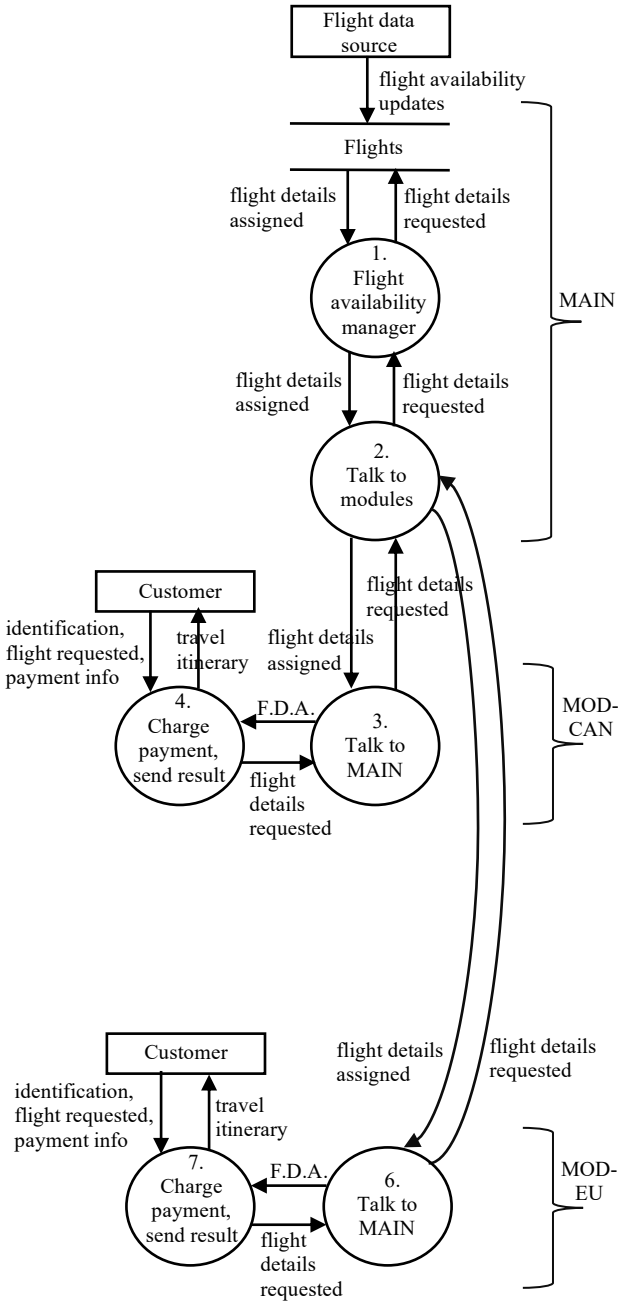


Figure 8. DFD for ACCURES after changes, "F.D.A." stands for "Flight details assigned" – used due to lack of space. All data are SD.

It is possible to further reduce the attack surface of Figure 8, as follows:

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

- Using method c), deny an attacker access to the SD link between process 3 and 4 by implementing MOD-CAN on one physical platform (see explanation for method c) above). Similarly, deny access to the SD link between processes 6 and 7 by implementing MOD-EU on one physical platform. Finally, deny attacker access to the SD link between the Flights datastore and process 1 and the SD link between processes 1 and 2 by implementing MAIN on one physical platform. These changes were not seen as impacting performance or maintenance and were accepted.
- Using method a), obfuscate the Flights data store. Next, obfuscate the 5 SD links that extend outside the modules. These links are: between "Flight data source" and the Flights datastore, between process 2 and process 3, between process 2 and process 6, between "Customer" and process 4, and between "Customer" and process 7. These changes were deemed feasible. Consideration was also given to obfuscating or combining some of the processes, but this was not done as it would significantly impact performance.

These changes do not alter the DFD in Figure 8. Table VI gives the remaining attacker accessible SD locations. The new attack surface is of size $N = m + 2n + k = 0 + 0 + 6 = 6$. The application of the methods in Section III-A has improved the security of sensitive data in the system by reducing the size of the attack surface from 29 to 6.

TABLE VI. REMAINING ATTACKER ACCESSIBLE SD LOCATIONS

	Links	Datastores	Processes
1			process 1
2			process 2
3			process 3
4			process 4
5			process 6
6			Process 7

V. RELATED WORK

In this section, we refer to "attack surface" in the general sense.

Most closely related to this work is this author's previous work on reducing the attack surface [9][10]. However, this previous work differs from the current work in at least the following two ways: a) the previous work proposes a graphical model with which to identify the attack surface whereas the current work does not require any such model, and b) the previous work reduces the attack surface by requiring the developer to learn and modify the graphical model whereas the current work has no such requirement.

Some of the following related works deal with attack surface identification and reduction at the code or binary levels, whereas this work deals with it at the architectural level. A few of these works reduce the attack surface by removing unnecessary code or features similar to the removal of SD locations in this work. A. Kurmus et al. [11] look at reducing the attack surface of commodity OS kernels by identifying code that is not used and removing it or preventing it from executing. T. Kroes et al. [12] investigate reducing the attack surface through dynamic binary lifting, removal of unnecessary features, and recompilation. R. Ando [13] presents work on attack surface reduction through call graph enumeration in which attackable call graphs are removed. S. N. Bukhari et al. [14] propose reducing the attack surface corresponding to cross-site scripting by employing secure coding practices. G.V. Neville-Neil [15] writes that "the best way to reduce the attack surface of a piece of software is to remove any unnecessary code". Obermeier et al. [16] propose to reduce the attack surface of next-generation industrial control systems through the use of a dynamic security system that adapts the parameters of network and security controls according to underlying changes in the control system environment. This results in the security controls only allowing data transfer that is required by the control system, thus reducing the attack surface.

The following works look only at identifying the attack surface. M. Sherman [17] investigates attack surfaces for mobile devices. This author claims that mobile devices exhibit attack surfaces in capabilities, such as communication, computation, and sensors, that are generally not considered in current secure coding recommendations. C. Theisen et al. [18] propose the use of risk-based attack surface approximation (RASA) which uses crash dump stack traces to predict what code may contain attackable vulnerabilities. Their goal is to help software developers prioritize their security efforts by providing them with an attack surface approximation. P. K. Manadhata and J. M. Wing [5] provide a much more detailed metric of attack surface than is defined in this work. Their metric may be considered as a measure of the size of the attack surface for all possible threats. Our definition of data attack surface may be considered as a subset of their definition. It is not possible to compare these two definitions in terms of accuracy or usefulness since they were defined with different purposes in mind. The same applies to any attempt to compare our definition with any other definition of attack surface.

Software attack surface identification and reduction is closely related to software vulnerability analysis, where the greater part of research also appears to be at the code level. Perl et al. [19] use an SVM classifier to find vulnerabilities in code repositories. Li et al. [20] describe VulPecker, a tool that can find a specific vulnerability in source code. Pang et al. [21] describe predicting vulnerable software components using a method built on a deep neural network. Anand et al. [22] suggest a way of classifying security patterns based on the type of vulnerability they treat. Also in this vulnerabilities category but working at the architectural level is this author's work, Yee [23], which deals with using a graphical model to identify and remove vulnerabilities during design. Yee [23] differs mainly from the current work in that it focuses on vulnerabilities found through risk analysis whereas the current work focuses on the data attack surface found by counting the attacker accessible SD locations in a model of the system such as its DFD.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Some works propose to increase security through attack surface expansion rather than attack surface reduction. For cloud services, T. Al-Salah et al. [24] propose three attack surface expansion approaches that use decoy virtual machines co-existing with the real virtual machines in the same physical host. They claim that simulation shows that adding the decoy virtual machines can significantly reduce the attackers' success rate. For enterprise networks, K. Sun and S. Jajodia [25] propose a new mechanism that expands the attack surface, so that attackers have difficulty in identifying the real attack surface from the much larger expanded attack surface. Note that these works do not contradict reducing the attack surface to improve security, since the attack surface is not really expanded but only appears to be expanded due to the addition of decoys.

VI. Discussion

It has been suggested that combining multiple SD locations into a single SD location using attack surface reduction method b) will result in a greater loss should the attacker target this combined location. However, recall that all SD locations on the data attack surface are attacker accessible. Therefore, the multiple locations existing prior to combining were all attacker accessible. But because of the greater attack surface prior to combining, it is more likely that all of those locations will be attacked than the single combined location. Note that prior to combining, the attacker will attack all the locations rather just a few, since there is no reason for him/her to stop until he/she gets all the SD. Thus, the likelihood of the single combined location being attacked after combining is less than the likelihood of the multiple locations being attacked prior to combining, and the potential data loss is the same both after combining and prior to combining. This shows the advantage of a smaller attack surface. In addition, there is always the possibility of applying reduction method a) (make a SD location useless to the attacker) to the combined location, removing it from the data attack surface.

Definition 3 describes the attacker accessible SD locations that make up the data attack surface as containing SD that is in the clear. It has been suggested that these locations can also contain encrypted SD since some attackers may still attack such locations, in order to obtain encrypted SD which they would then decrypt. While this is possible, our view in this work is that attackers would not attack such locations, since there are many other attacker accessible SD locations that contain SD in the clear. However, if we were to allow locations to contain encrypted SD, our attack surface reduction approach would be impacted as follows: reduction method a) would not be applicable to such locations, and such locations would remain part of the data attack surface, unless eliminated by methods b), c), or d). The overall impact would be slightly fewer opportunities to reduce the data attack surface.

We admit that our choice of $M_l = M_p = 1$ and $M_d = 2$ in Section II-B may not be accurate, but accuracy is not needed to show that the data attack surface is smaller after each reduction method is used. Nevertheless, these multiplier values do represent a reasonable approximation to the real values, which can only be ascertained with further research studies. Furthermore, perhaps the real values are not so important, since the true benefit of knowing the size of the data attack surface is to be able to use it for comparison and measurement purposes as a result of some improvement effort, i.e., the capabilities listed in the first paragraph of Section II-B, and our values already provide this benefit.

VII. Conclusions and Future Work

This work has presented an easy way to identify and calculate the size of the attack surface for sensitive data held within an online computer system, based on finding attacker reachable SD locations in the system. This work has also introduced methods for reducing the attack surface that are to be applied at the architectural level early in the development cycle, prior to coding, as part of the Design for Security toolset.

Applying the methods does not require developers to learn a new model or a new coding language. Apart from the methods themselves, which are straightforward, a minimal level of security knowledge is needed, in order to understand the concept of data attack surface, the purpose of the methods, and how they work. Knowledge of the computer system is the major requirement, but developers already have this knowledge. Although the methods themselves are straightforward, applying them can be challenging in terms of their impact on performance, ease of maintenance, and other factors, as mentioned above. However, the goal of applying the methods is not to obtain the smallest attack surface possible, but rather to reduce the attack surface while balancing the needs of performance, reliability, maintainability, and so on. Thus, it is quite acceptable not to have attained the smallest attack surface possible, so long as those other needs are satisfied. We expect the methods to be acceptable to developers and their management because of their practicality and ease of application.

Future work includes improving the identification of the attack surface and the calculation of its size. In addition, we hope to refine the methods for reducing the attack surface from developer feedback, obtained perhaps through workshops and trials. Other future work consists of investigating new methods for reducing the attack surface and looking at tools that could indicate a method's impact on such aspects as performance, reliability, ease of maintenance, and implementation costs.

References

[1] G. Yee, "Reducing the Attack Surface for Private Data," Proc. Thirteenth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2019), October 2019, pp. 28-34.

[2] Identity Force, "2020 Data Breaches – The Worst So Far," [retrieved: December, 2020] https://www.identityforce.com/blog/2020-data-breaches

[3] G. Yee, "Visualization and Prioritization of Privacy Risks in Software Systems," International Journal on Advances in Security, issn 1942-2636, vol. 10, no. 1&2, pp. 14-25, 2017, [retrieved: Dec., 2020] http://www.iariajournals.org/security/

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[4] C. Salter, O. Sami Saydjari, B. Schneier, and J. Wallner, "Towards a Secure System Engineering Methodology," Proceedings of New Security Paradigms Workshop, Sept. 1998, pp. 2-10.

[5] P. K. Manadhata and J. M. Wing, "An Attack Surface Metric," IEEE Transactions on Software Engineering, vol. 37, no. 3, pp. 371-386, May/June, 2011.

[6] M. L. Collins, M. C. Theis, R. F. Trzeciak, J. R. Strozer, J. W. Clark, D. L. Costa, T. Cassidy, M. J. Albrethsen, and A. P. Moore, "Common Sense Guide to Mitigating Insider Threats, Fifth Edition," Software Engineering Institute, Report Number CMU/SEI-2016-TR-015, December 2016.

[7] D. M. Cappelli, A. P. Moore, and R. F. Trzeciak, The CERT Guide to Insider Threats, Addison-Wesley Professional, ISBN 9780321812575, January 2012.

[8] T. DeMarco, Structured Analysis and System Specification, Prentice Hall, May, 1979.

[9] G. Yee, "Modeling and Reducing the Attack Surface in Software Systems," Proceedings, 11th Workshop on Modelling in Software Engineering (MiSE'2019), May 2019, pp. 55-62.

[10] G. Yee, "Attack Surface Identification and Reduction Model Applied in Scrum," Proceedings, 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), June 2019, pp. 1-8.

[11] A. Kurmus, A. Sorniotti, and R. Kapitza, "Attack Surface Reduction for Commodity OS Kernels: Trimmed Garden Plants May Attract Less Bugs," Proceedings of the Fourth European Workshop on System Security (EUROSEC '11), April 2011, article no. 6 (no page number available).

[12] T. Kroes, A. Altinay, J. Nash, Y. Na, and S. Volckaert, "BinRec: Attack Surface Reduction Through Dynamic Binary Recovery," Proceedings of the 2018 Workshop on Forming an Ecosystem Around Software Transformation (FEAST '18), October 2018, pp. 8-13.

[13] R. Ando, "Automated Reduction of Attack Surface Using Call Graph Enumeration," Proceedings of the 2018 2nd International Conference on Management Engineering, Software Engineering and Service Sciences (ICMSS 2018), January 2018, pp. 118-121.

[14] S. N. Bukhari, M. A. Dar, and U. Iqbal, "Reducing Attack Surface Corresponding to Type 1 Cross-Site Scripting Attacks Using Secure Development Life Cycle Practices," Proceedings of the 4th International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB-18), February 2018, pp. 1-4.

[15] G. V. Neville-Neil, "Reducing the Attack Surface," Communications of the ACM, vol. 61, issue 2, pp. 27-28, February 2018.

[16] S. Obermeier, M. Wahler, T. Sivanthi, R. Schlegel, and A. Monot, "Automatic Attack Surface Reduction in Next-Generation Industrial Control Systems," Proceedings of the 2014 IEEE Symposium on Computational Intelligence in Cyber Security (CICS), December 2014, pp. 1-8.

[17] M. Sherman, "Attack Surfaces for Mobile Devices," Proceedings of the 2nd International Workshop on Software Development Lifecycle for Mobile (DeMobile 2014), November 2014, pp. 5-8.

[18] C. Theisen, B. Murphy, K. Herzig, and L. Williams, "Risk-Based Attack Surface Approximation: How Much Data is Enough?," Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP '17), May 2017, pp. 273-282.

[19] H. Perl, S. Dechand, M. Smith, D. Arp, F. Yamaguchi, K. Rieck, S. Fahl, and Y. Acar, "VCCFinder: Finding Potential Vulnerabilities in Open-Source Projects to Assist Code Audits," Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS'15), October 2015, pp. 426-437.

[20] Z. Li, D. Zou, S. Xu, H. Jin, H. Qi, and J. Hu, "VulPecker: An Automated Vulnerability Detection System Based on Code Similarity Analysis," Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC'16), December 2016, pp. 201-213.

[21] Y. Pang, X. Xue, and H. Wang, "Predicting Vulnerable Software Components through Deep Neural Network," Proceedings of the 2017 International Conference on Deep Learning Technologies (ICDLT'17), June 2017, pp. 6-10.

[22] P. Anand, J. Ryoo, and R. Kazman, "Vulnerability-based Security Pattern Categorization in Search of Missing Patterns," Proceedings of the 2014 Ninth International Conference on Availability, Reliability and Security (ARES), September 2014, pp. 476-483.

[23] G. Yee, "Removing Software Vulnerabilities During Design," Proceedings of the 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol. 2, July 2018, pp. 504-509.

[24] T. Al-Salah, L. Hong, and S. Shetty, "Attack Surface Expansion Using Decoys to Protect Virtualized Infrastructure," Proceedings of the 2017 IEEE International Conference on Edge Computing (EDGE), June 2017, pp. 216-219.

[25] K. Sun and S. Jajodia, "Protecting Enterprise Networks through Attack Surface Expansion," Proceedings of the 2014 Workshop on Cyber Security Analytics, Intelligence and Automation (SafeConfig '14), November 2014, pp. 29-32.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Applicability of a Cryptographic Metric Taxonomy in Cryptosystem Procurement Process and in Evaluation of Open Standards

Outi-Marja Latvala*, Jani Suomalainen†, Kimmo Halunen*,
Markku Kylänpää†, Reijo Savola*
VTT Technical Research Centre of Finland
*Oulu and †Espoo, Finland
{firstname.lastname}@vtt.fi

Mikko Kiviharju
Finnish Defence Research Agency
Finland
mikko.kiviharju@mil.fi

***Abstract*—Measuring the security of cryptographic systems in a simple and effective way is a difficult problem. There are several metrics that need to be taken into account. Earlier studies have produced one taxonomy of these different metrics, but the applicability of the taxonomy and the different metrics have not been tested. In this paper, we present a revised taxonomy of metrics for cryptographic systems and show results of applying it in two different scenarios: a procurement process for cryptosystems and in evaluation of open standards, namely the TLS 1.2 and TLS 1.3 standards. Applicability and meaningfulness of a taxonomy depends on its ability to differentiate cryptosystems and thus enable comparisons. Our results show that the revised taxonomy can help in differentiating systems and standards, especially when examining implementation related metrics. Future work should streamline the overly complex evaluation process.**

Keywords–cryptography; metrics; taxonomy; evaluation

I. INTRODUCTION

Measuring the security of systems against adversarial attacks is a very difficult problem. In cryptography, there exist some measures for the security of cryptosystems, but a comprehensive measure is still lacking. In [1] we presented a first version of a taxonomy of different metrics of cryptographic systems. In this paper, we extend that work and provide a revised taxonomy. This work is based on applications of the metric in some use cases by measuring the cryptosystems with the help of the metric taxonomy.

A good test for a metric is its applicability in real world use cases. Cryptosystems are used in many different products and protocols in modern society. Some areas, where the security of cryptosystems is crucial, include government communications (military and diplomatic use) and banking. In order for a cryptographic product to be used in sensitive government applications, cryptosystems and applications need to be certified. The certification is usually a fairly lengthy process especially when higher confidentiality/classification levels are aspired for.

Cryptosystems are needed and used also in our everyday communications and digital services. Without the many advances in cryptography, it would be extremely difficult to build digital services at the scale we are seeing now. Especially public key cryptography has played a crucial role in this development [2]. Thus, measuring also the security of the cryptosystems that secure these communications and services is important both to assure their trustworthiness as well as to enable their continuous improvement.

In order to better understand different metrics and the attributes that they measure, we need to have a taxonomy of these. Existing efforts for providing taxonomies for cryptographic metrics include e.g., Benenson et al. [3] who explored metrics from attackers' point of view and Jorstad et al. [4] who studied metrics in algorithms. Several standardization efforts, e.g., [5]–[8], have also provided guidelines or criteria for implementations. To the best of our knowledge, a comprehensive view of cryptographic metrics has been lacking until recently. The previous work of [1] provided a comprehensive taxonomy of metrics that gave concrete and generic measures both for algorithms as well as for implementations. This paper revises the taxonomy. The applicability of the metrics is increased as standards and products can be distinguished from each other in more accuracy. The revised taxonomy also provides new details and clarifies definitions.

We provide results from applying this metric to six real world use cases related to cryptographic products offering communications confidentiality services. We also apply our taxonomy in evaluating two different versions of the Transmission Layer Protocol (TLS). The results of these case studies are described and analysed. The revision of the taxonomy is based on lessons learned from these case studies.

The paper is organised in the following way. The next section describes the background for our work including an overview of the previous version of the taxonomy. The third section is dedicated to our case studies that provide the rationale for further improving the taxonomy and metrics. The fourth section describes our revised taxonomy and fifth section discusses our findings. Finally, we give conclusions from our research and some future directions for further study.

II. BACKGROUND

In this section, we present the relevant background on measuring cryptography and on the certification of cryptosystems for classified communications.

This research has been funded by Defence Forces Research Program 2017 (PVTO 2017).

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

A. Measuring Security

It is noteworthy, that there are many measures and metrics for security in general and also for cybersecurity specifically. These can take many forms and have a number of different dimensions. These also work on several levels of abstraction ranging from a cybersecurity index measuring nations [9] to a measurement of a single device or a piece of software (e.g., IoT labels [10]). However, measuring the security of cryptosystems does not have that many good metrics.

The terms security performance and level are commonly used in practice to refer to the *effectiveness* of security countermeasures, the main objective of security work and solutions. In addition, *efficiency* is essential because personnel and time resources, and costs have always constraints. In addition to effectiveness and efficiency, *correctness* is a fundamental objective in security measurements. Correctness is a necessary but not sufficient requirement for effectiveness; i.e., it is a side effect in effectiveness, not its driver. Sufficiently effective security countermeasures are based on sufficient risk awareness of the target system in focus. There are various factors which enable effectiveness in practice, such as risk-driven design of security controls, configuration correctness, sufficiently rigorous implementation and deployment of security controls and proper security assurance activities. In practice, complexity, limited observability, a lack of common definitions and the difficulty of predicting security risks make it impossible to measure security as a universal property. Therefore, the term security metrics is misleading, yet widely used [11].

In practice, there are various gaps and biases between security effectiveness measurement objectives and practical security measurements. In risk analyses, it is not possible to identify and prioritize all actual risks. Difficulties in understanding well the target system or risk situation can cause bias. Further gaps are introduced when developing security requirements and the actual system. Different phases of R&D cause easily additional bias. Due to the gaps and biases, security effectiveness can be achieved only asymptotically. In security metrics modelling, an important goal is to minimize gaps and biases, making the more practical security correctness objectives as close as possible to security effectiveness objectives [11].

According to the results of the expert opinion survey reported in [12], *correctness, measurability* and *meaningfulness* are the core quality criteria for security metrics. Moreover, usability is considered to be important, but not so essential as the three before-mentioned dimensions. In the following, we discuss these quality dimensions and their relationships to more detailed quality criteria.

The term accuracy is often used instead of correctness to emphasize the fact that all-inclusive correctness is impossible. Using the term correctness makes it possible to differentiate between accuracy and precision. The time dependability of metrics can be seen partly as a sub-criterion of prediction accuracy and partly as an independent sub-criterion of 'general' correctness. The representativeness of security metrics is crucial to their correctness in a security context. The granularity of a metric and associated measurement should be at least at a level where adequate decision-making based on them is possible. Contextual specificity (or, inversely, contextual independence) is a special case of granularity. Completeness of security metrics is related to representativeness, addressing one or a collection of metrics. Non-intrusiveness is an important criterion related to correctness. Security measurements and the associated metrics should not overly affect or hinder the actual functionality of software systems or the functions of an organization [12].

Measurability is a prerequisite for the meaningfulness and usability of security metrics. Attainability of measurable information is related to measurability, while availability is a sub-goal of attainability. Together, reproducibility and repeatability form the precision of the measurement. Scale reliability is a sub-criterion of reproducibility. In addition to measurability, reproducibility, repeatability, and scale reliability are related to correctness.

To be meaningful, the metric should answer the original essential question that reflects the need for evidence. Clarity is closely related to meaningfulness: the clearer the formulation of the metric, the easier it is to understand provided that the person interpreting it has enough knowledge about the underlying context. Succinctness increases clarity and thereby meaningfulness. Good succinctness also increases the efficiency of the metric, a sub-goal of usability. In order for the security metric to be meaningful, they should incorporate applicability to decision-making. Comparability of different measurement results is desirable when making selection decisions among different security controls. The ability to show progression, is a special case of comparability [12].

Usability of security metrics is important, yet not as critical as correctness, measurability and meaningfulness, because poor usability is not fatal for security metrics. The criteria related to usability include the following: efficiency, cost effectiveness and controllability, scalability, and portability [12].

B. Measuring Cryptography

For many years, the most prevalent measure used in describing the security of cryptographic systems has been the algorithm-specific key length, with recommendations from governmental authorities as well as standardization bodies [13], [14]. There are, however, a number of shortcomings when using only this one metric to evaluate cryptographic systems: firstly, the update cycle of the recommendations may be years, and the coverage of standards only includes the most commonly used systems. Secondly, the key length indicates resilience in the most simplistic adversarial models only: the so-called "brute-force" attack, where the attacker only tries to guess (or compute) the key. This leaves out multiple implementation-level issues [15], protocol vulnerabilities, some more niche use cases [16] of regular algorithms and many more.

There are also cases, when new algorithms are evaluated for completely new applications that are not covered by current standards; in these cases, it would be preferable to have more general set of metrics to use. Finally, there are many different levels of abstraction and each level combines methods from lower levels to reach the security goal of that level. Even when a security proof exists and gives a good guideline on how to choose the security parameter, this one parameter - the key length - oversimplifies the complexities of implementing

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

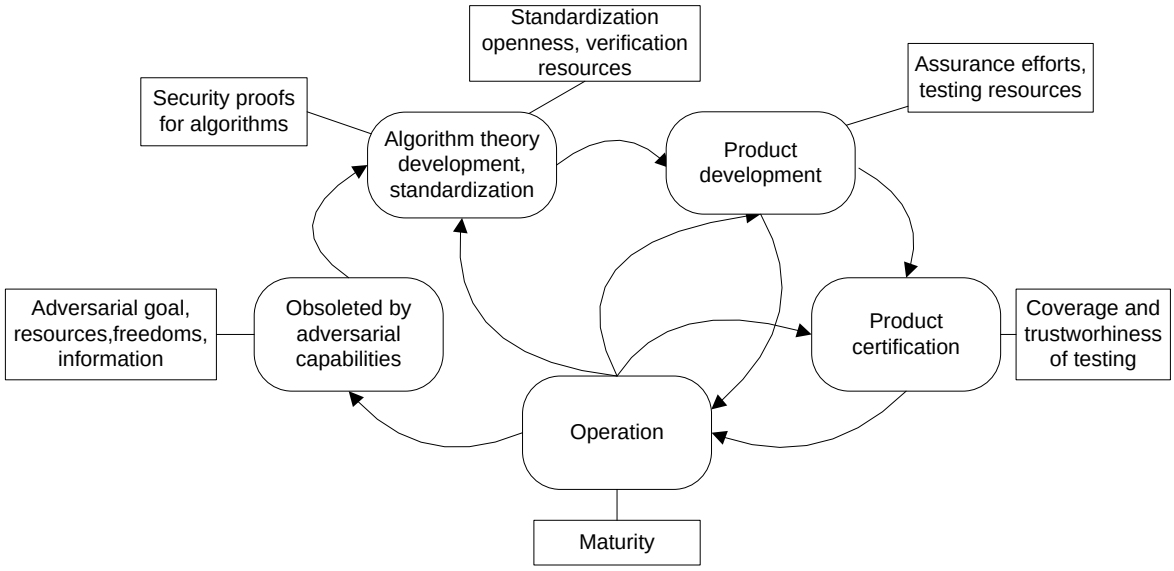


Figure 1. A life cycle for cryptographic systems and its relation to metrics.

cryptosystems. In mathematical terms, a certain key length can be seen as a necessary, but not sufficient condition for security of some cryptosystem in some given context, e.g., a digital signature protocol.

One of the main uses of metrics is to compare candidate cryptosystems for a specific purpose. In this case, it may happen that the security features are all approximately on the same line, but due to some conventions in the detailed implementation, they do not perform equally well. In this case, performance-related metrics become relevant, and we claim that there are certain performance-related metrics that are fairly common across the use of cryptographic primitives.

Metrics can be identified from different phases of the life cycle of cryptographic systems (illustrated in Figure 1). Initiation of development of new cryptography is typically motivated by expanding adversarial capabilities, which are measured through resources, goals, freedoms of action and information. Acceptance of new algorithms is achieved with theoretical security proofs as well as through standardization by openness and verification. During the algorithm development and standardization, developers and academic reviewers verify and analyze the theoretical strength of algorithms using common metrics and methodologies. These may be specific to the use case or more generic measures accepted in the community for measuring the cryptosystem.

For example, with hash functions there exist many different results on potential attacks on the Merkle-Damgård type iterative hash functions. However, not all these have been considered serious enough to be explicitly defended against for example in the NIST call for candidates for the SHA-3 hash function standard. This shows that there exists a some sort of a consensus on what is considered a serious threat to the security and what does not among the community. In some cases, the results that show an attack can be derivative of more fundamental attacks or properties of the system. In the case of hash functions for example multicollision attacks result from the underlying property of iteration irregardless of the implementation [17], [18]. Some of these have been generalised also to the sponge construction employed in the current SHA-3 standard [19].

During the implementation of software or hardware products, the vendors test products to detect implementation failures. Developers also often expose products for certification testing that verifies that the product fulfils the intended security criteria. Users may also require certification before accepting products for procurement and operational use. Independent or national accredited testing laboratories have product certification frameworks and programs for assuring that implementations have required functionality and behave as expected with different inputs. During deployment and operation, the product matures and its parameters must be adjusted to withstand threats from evolving environment. Technologies and implementations mature within time until new disruptive adversarial capabilities or new security requirements necessitate new algorithms and products. The legacy systems and algorithms are phased out over time. Changes in operational context - new performance requirements or needs to certify products against emerging threats - may also result in changing the phase of the cryptographic system in the life cycle (illustrated with arrows in Figure 1).

C. Certification of Cryptosystems

An important area, where metrics for cryptosystems are needed, is the certification (or approval) of cryptographic products. The need for certification exists in many different areas of official use. The process of certification can be very cumbersome and is based on evaluations. These evaluations can be very lengthy and be based on some standards (e.g., Common Criteria [20]) or some more heuristic criteria set by the certifying authority.

A comprehensive, applicable and possibly even simple measure for the strength of a cryptosystem would make these evaluations easier to conduct and faster. It could also help in comparing cryptosystems designed for different purposes and

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

systems certified under differing and proprietary standards. A good metric could also be used as a guideline or threshold for cryptosystems to be used for certain levels of classification.

Governments routinely use a well-defined evaluation and approval process to ascertain that systems used for protecting resources on a certain sensitivity level are actually fit for the purpose. Due to the nature and use of cryptologic information (i.e., for intelligence gathering), the details of these processes are often hidden and proprietary. The usual outcome of such a process merely answers *"yes"/"no"* for questions of the type: *"Is product X fit for protecting assets on sensitivity level Y?"*. Thus, it is meaningless to compare two products on the same level Y, which result from different certification bodies. This also implies the need for standard metrics.

In high-security environments there may also be a disconnect between procurement and assurance processes. Then it is not even expected that the Request for Information (RFI) or Request for Quotation (RFQ) phases produce information related to security metrics (other than what can be gleaned from open sources and standards). Security metrics are then evaluated with a pass/fail-grade just before actual procurement. This type of operation is deemed necessary in environments where there are contradicting requirements of secrecy and information needs, and may result in sub-optimal outcomes for the procurer, when compared to a completely open process.

There are several standards that have been developed to support certification of cryptosystems. For example, several ISO/IEC standards [5], [6], [21] consider the requirements of cryptographic systems, the verification and testing of such systems etc. These give good guidelines for certification, but do not in general define specific metrics and values that need to be met and measured.

D. Previous Taxonomy for Measuring Cryptosystems

The first comprehensive taxonomy for measuring cryptosystems in [1] classified metrics into four main categories. The first two categories - adversarial model and security proof framework metrics - addressed metrics that are related to security of the algorithms. The latter two categories - verification and maturity as well as cost and performance metrics - addressed feasibility and security of cryptographic implementations.

The authors of the previous taxonomy defined a cryptosystem to be any algorithm, protocol or method for providing cryptographic guarantees with respect to some security goal. They also defined *a metric* as a way to measure some part or the totality of the security of a cryptosystem. A metric can have numerical values or it can be a qualitative description depending on the use case. A metric is *measurable* if there is a standard convention on how the metric is measured and this is uniform across all applications of the metric (e.g., kilograms for weight). A metric is *semi-measurable* if there are several different conventions on how to measure the metric and some of these are not readily comparable with each other. The third category of metrics is *non-measurable*, which means that a standard for measurement does not exist or that the different values that the metric can have are not comparable in meaningful ways. Note the difference to the mathematical concept of a metric, which always has a numerical value. We also distinguish between *quantitative* and *qualitative* metrics. Quantitative metrics give a numerical or several numerical values to the cryptosystem and qualitative metrics give a description of the state of the cryptosystem.

In the original taxonomy, also the practical relevance was represented by stating whether an attribute was theoretical or practical. In our revised taxonomy, we have chosen not to use this attribute. We think that this distinction is not meaningful, as the metrics need to be actionable at all levels of abstraction. However, it is possible that for a certain use case and scenario, some metrics can have greater practical impact than others.

One of the major shortcomings of the previous taxonomy is that in practical applications, many cryptosystems utilise certain standards (e.g., TLS) and thus the values of metrics of many implementations do not vary in meaningful ways. For example, the adversarial model metrics are usually fixed in any standard. Thus, these metrics cannot differentiate between different cryptosystems against *the same adversarial model*. Thus, there is a need for metrics that can provide the differentiating factors.

On the other hand, it is important to measure also the adversarial model metrics so that we can distinguish different levels of security between different adversarial models. This can be useful not only for practical implementations of cryptosystems but also in research. By finding differences in the metrics, one can find room for improvement and new research problems in trying to address these.

III. Revised Taxonomy

Our taxonomy, which is presented in Table I, has three main categories. The first two relate to algorithmic metrics that involve cryptosystems independently of their realization in code or hardware. Algorithmic metrics are here divided to adversarial model metrics (Subsection III-A) and proof framework metrics (Subsection III-B). The last category combines different metrics that are relevant for evaluating realizations and implementations of cryptosystems, their feasibility, maturity, and costs (Subsection III-C).

We revised the original taxonomy from [1]. The main difference is that in the highest categorization level, we now have combined all cryptosystem implementation related metrics under the same main category - verification and feasibility - as the methodologies for verifying various metrics are similar. Refined metric taxonomy contains also some new metrics and definitions for old metrics have been clarified based on feedback and experience on applying the taxonomy.

One issue in the old taxonomy was the disconnect between theoretical adversarial models and the implementation level differences in actual threats and capabilities of adversaries. For example, one could have a disk encryption application (e.g., Bitlocker [22]) and a TLS implementation using AES with the same keylength. In a theoretical sense and in the metric of [1] these would yield the same result in the metric. On the practical and implementation level, there are many differences on the information that an attacker can get on the data and the overall threat model. These are not reflected well in the original taxonomy. Thus, we have added new metrics to the taxonomy. These are *Side-channels* and *Metadata*.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE I. CATEGORIES AND PROPERTIES OF DIFFERENT METRICS OF CRYPTOGRAPHIC SYSTEMS.

Main category	Subcategory	Metric	Measurable	Type
Adversarial model	Degrees of freedom	Observe/choose/choose adaptively	yes	Qualitative
		Corruption power - num. of principals	yes	Quantitative
		Corruption power - degree of corruption	semi	Qualitative
		Security game compliance	semi	Qualitative
	Adversarial available information	Pre-crypto	yes	Quantitative
		Post-crypto	yes	Quantitative
		Secret key material	yes	Quantitative
		Protocol runs	semi	Quantitative
		Setup parameters	semi	Qualitative
		Simulation environment	semi	Qualitative
	Adversarial goal	Goal	semi	Qualitative
	Adversarial resources	Computation power Instantiated	yes	Quantitative
		Computation power Non-instantiated	semi	Qualitative
		Memory Instantiated	yes	Quantitative
		Memory Non-instantiated	semi	Qualitative
Proof framework	Security assumptions	Mathematical complexity	semi	Qualitative
	Abstraction assumptions	Type	semi	Qualitative
		Num. of assumptions	yes	Quantitative
		Maturity of assumptions	no	Qualitative
	Methodology	Tightness	yes	Quantitative
	Rigor	Rigor	semi	Qualitative
Verification and feasibility	Assurance levels	Assurance standard or profile	no	Qualitative
		Evaluation assurance level	yes	Quantitative
	Test coverage	Percentage of covered areas	yes	Quantitative
	Vulnerabilities	Number	yes	Quantitative
		Number of classified	semi	Quantitative
	Side-channels	Existence	semi	Qualitative
	Metadata	Leaked amount and type	no	Qualitative
	Evaluator acceptance and reputation	Reviews	yes	Qualitative
	Evaluator experience	Academic publications	semi	Quantitative
		Experience in years	yes	Quantitative
	Verification time	Time since released for evaluation	yes	Quantitative
		Size and efforts of eval. community	yes	Quantitative
	Openness of target	Software/design	semi	Qualitative
	Readiness level	Technology readiness level	yes	Quantitative
		Integration readiness level	yes	Quantitative
		System readiness level	yes	Quantitative
		PETS maturity model	yes	Qualitative
	Key length	Bits for criteria compliance	yes	Quantitative
	Time costs	Execution overhead	yes	Quantitative
		Communication overhead	yes	Quantitative
	Memory and transmission costs	Run-time memory	yes	Quantitative
		Storage capacity	yes	Quantitative
		Communication bandwidth	yes	Quantitative
	Implementation complexity	Size of software	semi	Qualitative
		Dedicated hardware requirements	semi	Qualitative
	Energy efficiency	Algorithm complexity dependent joules	yes	Quantitative
		Hardware platform dependent joules	yes	Quantitative

A. Adversarial Model Metrics

Cryptology and especially cryptographic theory aims to formalize how cryptographic algorithms work and withstand cryptanalysis. Due to the need for rigorous formalisms in cryptographic theory, the models used need to be very detailed, and yet general with respect to adversarial behaviour.

Algorithmic metrics are sometimes difficult to define and may be difficult to compare, due to the close association to actual schemes. Many qualities that are essential to one type of cryptographic algorithm may make no sense with another. However, there are still metrics that can be measured and that can be used to measure cryptosystems of similar nature (e.g., block ciphers or digital signatures). Some of these can then also generalise to metrics that can be used to wider varieties of cryptosystems.

For this purpose, even in the abstract world of algorithms and cryptography theory, it is beneficial to be able to state that algorithm A is (X times) more secure than algorithm B. This in turn requires some metrics, and to distinguish them from the other metrics used in this paper.

This results in the following:

- Algorithmic metrics are used in generally accepted combinations, rather than picked independently. In general, there exists good consensus which metrics form a good and reasonable combination. On the other hand, it is entirely possible that new cryptosystems will need to combine these metrics in new ways. Also, new metrics can be brought about through research.
- The exact definitions tend to be scheme specific, resulting in different understanding of which constructs or postulated goals are actually more secure than others, making some metrics effectively incomparable with each other.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

This is one of the main difficulties in measuring and comparing cryptosystems.
- Metrics themselves may be scheme and usage-specific: protocol security metrics may not apply to primitives, and asymmetric primitive security metrics not to symmetric primitives.
- Just collecting all the possible values of the metrics ever used, from thousands of papers is a daunting task, which is why we only give examples of the values. This means that a measurement using these metrics needs to be aware of the context and also the research on the topic.

As an example, consider the combination of the metrics in the following common concept: INDistinguishability under Chosen Ciphertext Attack or IND-CCA [23] and existential unforgeability under chosen-message attack, or EUF-CMA [24]. We observe here the following independent metrics:

- Adversarial goal: distinguish between random strings and actual ciphertext.
- Adversarially available information: a polynomial amount of information, before and after the cryptographic transformation.
- Adversarial degrees of freedom of actions include choosing the ciphertext-plaintext pairs adaptively (excluding the keys).

In addition to the three metrics above, we consider *adversarial resources*, which the designer of the cryptosystem expects the attackers to be able to wield. For existing attacks, the required adversarial resources may prove to be smaller than expected by the designers, which may break the system's security even in practice. The four above metrics together are related to the *adversarial model*. The metrics in the category are selected to be as independent of each other as possible. As mentioned already earlier, these dependencies are currently inescapable in the taxonomy work.

The adversarial resources consist of *computing power* and available *memory*. They are mostly well-defined and accessible metrics, with practical relevance.

Computing power is addressed here in both of its forms: exact attack complexities, and approximate or asymptotic complexities. Cryptographic theory rarely elaborates the adversarial models down to the detail of exact number of operations required to break the system. Instead, asymptotic estimates are given, and often even they are described only on the level of computational complexity classes.

In the case of exact complexities, values can be given, e.g., as the amount of floating point operations per second (FLOPS). In quantum computing, the unit can be based on, e.g., the amount of universal qubits and gates in the quantum computer or the quantum volume of the system [25]. As adversarial time is limited, the computing power is typically presented in relation to time, for example as CPU years (work done by 1 GFLOP machine in a year). The exact actual amount of computing required to break a cryptographic algorithm is known for demonstrated attacks. For example, finding SHA-1 collisions with 6500 CPU years or 100 GPU years [26], factoring of RSA-512 with 1.1 TFLOPS capacity in 4 hours [27], breaking of 7.45 DES keys within a nanosecond by custom ASICS hardware [28].

However, the exact work and time needed to break previously unbroken algorithms can be only estimated. Affecting adversarial factors include both the sophistication of breaking algorithms as well the type of processor {`CPU, GPU, ASIC, quantum`}. This metric is semi-measurable because for a given algorithm known attacks can be measured, but between different algorithms these might not be comparable. It is also important to note that for more complex cryptosystems breaking a single component of the protocol (e.g., a hash function) may not constitute a full breach of the system. This metric is also quantitative.

In the latter case, where the complexity class border is crossed, literature usually refers to different "computational models", the most common being probabilistic polynomial time (PPT) adversary corresponding to the complexity class bounded-error probabilistic polynomial time (BPP), where polynomially bound, probabilistic Turing machines are expected. Other notable models include bounded-error quantum polynomial-time (BQP) for quantum computers; and statistical, or unconditional security model, where the adversary is given limitless computational power. This metric is semi-measurable (as the exact relations between complexity classes are not known) and qualitative.

Because the exact running time estimates can only be fixed once a cryptosystem is fully instantiated and parametrized, we consider this measure to consist of two subclasses of the whole: instantiated and non-instantiated computing power (asymptotic notations can be computed to exact metrics once the parameters, such as key size, are fixed).

Memory is the amount of memory that the attack requires. Analogously to the computing power, we divide this into two subclasses: instantiated (measurable and quantitative) and non-instantiated (semi-measurable and qualitative). While not so common in cryptographic scheme design (with the exception of memory-hard password hashing), the attacks may require large memories expressible in complexity classes, e.g., with time-memory trade-offs. Also, the type of the memory and memory access can have effect on the time that the attack takes. Memory can also have some effect on the computing power needed for the attack. Some example memory complexity classes could be logarithmic space (LOGSPACE) and polynomial amount of space (PSPACE).

Adversarially available information is the amount and type of data that the attack needs or is allowed for the adversary. This can mean plaintext-ciphertext pairs, number of connections or interactions with a server, related keys, bits from a random generator, cryptographic parameters etc. Metric is quantitative within one type of data, but not necessarily across types, as this is scheme-dependent. Thus, the metric is only semi-measurable. We distinguish here at least six different types: *pre-crypto* (data before encryption, signing or other cryptographic transformation), *post-crypto, secret key-material* (symmetric or private asymmetric), *protocol runs, setup parameters* and *simulation environment master*. The four first ones are measured in bits, bytes or messages/keys/runs, the last two are discussed as follows:

- The access to setup parameters becomes relevant in cryptographic protocols, giving rise to, e.g., variants of Universal Composability (UC): Joint UC [29] and Global UC

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[30]. Possible value space could be {`local/global, per protocol/several runs`}.

- The control of the master process, which in cryptographic protocol security proofs models to what degree the adversary is able to control the (unspecified) protocol environment (resulting in yet other UC variants [31]). Possible value space could be {`Sim+Adv, Advonly, Env, *`}.

Technically, the scheme or protocol description and parameters of the system also belong to the measured adversarial information. However, modern systems very often assume all of them to be public, or constantly available (see Kerckhoffs's principle [32]), so we do not consider them here.

Adversarial goals are a complement to the security goals: if the security goal is indistinguishability, the corresponding adversarial goal is to distinguish (an output of cryptographic transformation from a random string); if the adversarial goal is a total break of the system, it suffices as a security goal to be able to keep even one message confidential. Thus, these two metrics are actually two sides of the same metric. Unfortunately, to be comparable, the goals need to be rigorously formalized, which usually results in case-specific definitions, and almost all values for the metric are incomparable, making it both qualitative and semi-measurable only. An example value space for typical goals is {`Semantic deduction, Information Leak, Local deduction, Global deduction, Total Break`}.

Adversarial degrees of freedom of action refer here to what the adversarial model is expecting the adversary to do. (Note: "anything" is not a rigorous enough answer to this). We propose to divide the degrees of freedom into three: *General, Corruption power* and *Game compliance*.

Corruption power. In interactive protocols, the adversary is also assumed to be able to access and/or modify the private information of some of the principals. This is called corruption, and depending on the scheme, only a certain number of principals are allowed to be corrupted. Sometimes even more fine-grained "corruptive power" is allowed [33]. The example values of this metric could include a (quantitative) percentage of corrupted principals and a (qualitative) description of the degree of corruption within one principal (see [33] for subprotocol-level detail).

For the *general metrics*, cryptographic formalisms differ in the amount of principals: Single-party settings (conventional encryption and signatures) and multiparty settings (protocols). As we show below, the multi-party setting does not bring that many new metrics per sé.

In the single-party setting, only one or fixed, integral set of cryptographic transformations (a black box) are usually considered. Thus, there is a set of black boxes (e.g., encryption and decryption, or signing and verification), with a number of inputs and outputs. In this case, the adversary may be able to *observe* some or all of the inputs, or to *choose* (possibly adaptively) some or all of them. Note that we consider modification of inputs and other adversarially available information to belong to the "choosing" process. Some of the possible values in the single-party setting would then be `'Observe'`, `'Choose'` and `'Choose adaptively'`, in increasing order.

In the multi-party setting, i.e., protocols, the situation with adversarial behaviour appears at first sight to be more complex, as the security models are more varied. (Dolev-Yao model, Inexhaustible Interactive Turing Machines [34], Reactive Simulatability [35] and others [36]). In the Dolev-Yao model [37], the principle is that the "attacker carries the message", or that the adversary is free to read, modify, add and delete protocol messages and corrupt protocol principals (in effect stealing their private key material). Relaxations to this model include [38], where corruption does not include divulging private key material, which makes the corruption probabilistic.

However, the convention we made in the single-party setting already covers the deletion, modification and adding of protocol messages, global setup parameters modification and protocol environment control, since the ability to choose message (/parameters/environment properties) for a single party translates to all of the above. Furthermore, corruption of principals itself (note: this does not include the number of principals) can be thought of as a combination of adversarially available (keying) information and the ability to choose keying inputs to cryptographic transformations. We thus conclude that we have not identified more metrics from the multi-party setting.

Game compliance. Many of the formalisms in cryptographic security can be divided into two: game-based approaches and simulation-based approaches. Game-based approaches are basically a protocol, which try to model the adversarial behaviour in some commonly thought scenarios. These approaches result in efficient schemes that are relatively easy to prove secure. Simulation-based approaches try to enable showing security irrespective of the adversarial behaviour. The best the attacker can do, is to perform the idealized, non-cryptographic tasks assigned to replace cryptography in the simulation (SIM) model, since all of the cryptographic tasks are idealized to be functionally equivalent non-cryptographic ones, but performing the tasks by other means than cryptography. This is a very strong model, and not many constructions can be proven secure in this.

In the metrics, we consider this distinction to be an adversarial degree of freedom in the sense that the adversary is either constrained to follow some security game protocol, or not. Loosely speaking, the simulation-based security proofs also capture the adversary's actions within a Turing Machine, making the proof technique similar to that of reductionist proofs with complexity assumptions. The main difference in the proof technique is that the degrees of freedom are larger with the Turing Machine (basically Turing-Complete) than with a dedicated security game. The values could be, for example {`Game-App, Game-Gen, SIM`}, making a further distinction between general security games and very application-specific games.

B. Security Proof Framework Metrics

Proof framework is the framework in which the security proof is conducted. This includes multiple assumptions (for abstractions of certain functions and for the complexity of

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

several mathematical problems), the rigor used (ranging from heuristics to verified full proofs) and the proof methodology.

A metric clearly tied to the proof methodology is *tightness* of the proof. This concept indicates how exactly the resource needs for different phases of the proof are estimated. In "loose" proofs, polynomial reductions (without stating the actual degree of the reduction polynomial) and upper limits are common, whereas tighter proofs uses "less margin", resulting in more optimistic adversarial resources and ultimately in more efficient parameters for a scheme. This metric is measurable and quantitative, as typical asymptotical $O(f(n))$ expressions are used here.

Complexity assumptions are the foundation of many types of cryptographic proofs especially in the realm of computational security. They are assumptions on the hardness of different mathematical problems, usually that their time-complexity is superpolynomial in the security parameter. These assumptions can have several metrics:

- Assumption's time-complexity. This has relevance in many asymmetric schemes, since the complexity is not always exponential. The metric is measurable, quantitative and practical, as it directly affects key size. The metric is expressed with the Big-Oh-notation (e.g., $O(f(n))$).
- Assumption maturity (we consider this to be in the verification category and not elaborated more here)
- Type, if the assumption belongs to a known sequence of implications (e.g., Decisional Diffie-Hellman (DDH) $\Leftarrow$ Computational Diffie-Hellman (CDH) $\Leftarrow$ Discrete Log (DL) problem.) A possible common labelling borrows from the general ordering for several problems, where decisional problems (DDH) are usually easier than computational problems (CDH), and finally the primitive inversion problem (DL): {`decisional, computational/search, inversion`}. This metric is semi-measurable and qualitative.

Abstraction assumptions cover how much the proof methodology uses abstractions, what kind of type they present and their maturity. Typical abstractions give different functions as ideal oracles, the most famous probably being the Random Oracle Model (ROM, [39] with variations in [40] and [41]). Many other oracles exist as well, e.g., the Generic Group Model (GGM) [42], the Ideal Cipher Model (ICM) [43], and the Common reference string model [44]. Sometimes the oracles are implicit, such as the Dolev-Yao modelling on encryption operations, which are assumed to be secure. If no abstractions are used, the proof is said to be conducted in the Standard Model.

The actual metrics are proposed as follows:

- The number of abstractions used. For a proof in the standard model this would be zero. Different abstractions would be weighed differently depending on their maturity and suitability for the cryptosystem
- Assumption maturity (we consider this to be in the verification category and not elaborated more here)
- Type. Not all of the abstraction are equal, as there are some known relations among them (e.g., ICM and ROM have been proven equal in some cases [45]). We then postulate that like with the complexity assumptions, there is a common metric able to classify abstraction assumptions as well, but we leave it for future study.

Rigor refers to the level of detail of the proof, its compliance to commonly used proof techniques and the assurance in the validity of the proof. Many schemes outside the cryptologic community often rely on pure heuristics, others merely state that the scheme is essentially similar to an earlier scheme and overlook the security proof completely. Many other systems are too complex to contain fully rigorous proofs in single conference papers, making the authors only outline the proofs. Ideally, proofs should be fully detailed, and externally verified. The value space for this metric would then be {`Heuristic, Referenced, Outlined, Full, Verified`}.

C. Verification and Feasibility of Implementations

The strength and correctness as well as maturity and feasibility of cryptographic implementations can be verified with different verification methods and testing tools.

1) Verification Metrics: Verification is a process establishing security, correctness, compliance, or validation of cryptographic implementation. Verification metrics describe the coverage and effectiveness of the verification and testing actions that the cryptographic product has passed.

Assurance Levels are measurements indicating system's security compliance when compared against common or standard evaluation and testing requirements. For instance, Evaluation Assurance Level (EAL) is a seven point-scale metric used by the Common Criteria (CC) [46] security evaluation framework for implementations; Common Criteria's Protection Profile is simpler two point-scale (compliant/non-compliant) metric for specific product categories; Cryptographic Algorithm Validation Program (CAVP) [7] defines functional and statistical tests for algorithms with a two-point (pass-fail) scale; Cryptographic Module Validation Program (CMVP) [47] defines validation tests for hardware implementations in four point scale (i.e., FIPS 140-2 security levels); and ISO 29128 [48] Protocol Assurance Levels define requirements for the scope and automation of formal modelling and verification of cryptographic protocols.

In addition to the generic frameworks, there also exist frameworks that are specific for industry field or for an area of cryptography. For instance, the Payment Card Industry [8] has defined its own test requirements for two point scale evaluation of cryptographic hardware modules. National Institute of Standards and Technology (NIST) has specified [49] a large suite for *randomness* testing. Randomness sources are important for many cryptosystems as the keying usually requires some randomness to provide security. This attribute can have values like user-supplied, system supplied, user + system supplied (e.g., password and a salt) and verified randomness. Verified randomness can also be quantum randomness, but the extent to which this improves security of a system is not necessarily easy to quantify. Assurance levels are semi-measurable and quantitative metrics.

Testing coverage refers to the percentage of potentially vulnerable areas that are verified. Coverage is complete if every area, as specified, e.g., by a particular certification criteria or test set, is verified. The areas that can be tested include,

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

e.g., symmetric algorithms, asymmetric algorithms, key management, functionality, interfaces and protocols, randomness, susceptibility to side-channel and fault injection attacks, lifecycle, as well as susceptibility to physical tampering and to reversing of obfuscated functionality attacks. Existing test suites, validation program requirements or common criteria profiles can be utilized when estimating whether all relevant areas are included to verification and whether all tests for the relevant areas are executed. This metric of coverage is measurable and quantitative.

Existing systems have often known *vulnerabilities* of different impact and consequences. A most serious vulnerabilities are typically patched but due to costs or backward compatibility vulnerabilities with lower risks may remain. A straightforward quantitative metric is the amount of known all kinds of vulnerabilities. A little more sophisticated metric is to classify vulnerabilities according to their seriousness and count instances in each category, e.g., through the CVE [50] and CVSS systems [51].

One important implementation consideration, which is often covered in verification, is the existence of *side-channels*. These are possible information sources that come from outside the adversarial model. It is noteworthy that some adversarial models can include some side-channels and show that a system is resistant to these. However, not many systems can be shown to be resistant to all possible side channels.

Most common side-channels are timing and power consumption, but depending on the cryptosystem there might be others such as errors and faults, sound, heat etc. We distinguish between the different side channels and each type of side channel is its own attribute. This leads to a situation, where we need to have an attribute for ”other” side channels that are not listed in our taxonomy. This is because new side channels can be found later on.

Side-channels are semi-measurable and qualitative. Possible values for the side channel metric are:

- known side-channel attack,
- no known attacks,
- covered by the adversarial model
- not applicable

However, it is important to note that in many cases the applicability of different side-channels varies greatly. The not applicable value can also be used in case the evaluation is done on a theoretical algorithm without a specific implementation.

Another implementation consideration is the amount and type of *metadata* that the system allows to the potential attacker. This may be the length of the message, number of recipients, the algorithms used in the cryptosystem, programming languages and/or libraries used by the system. Also, timestamps, user names, location etc. fall into this category. This is a non-measurable and qualitative metric. This metric can contain a lot of information about the context, where the cryptosystem is applied. Thus, it is important to have an understanding what is relevant to the current analysis. This also makes it difficult to measure the different values against each other as the context matters greatly.

Capabilities of evaluating laboratories, communities, and individuals - skills, experience, and methods - are an indirect measure also for the evaluated systems. The more skilled and scrutinized reviews the product has passed, the more less likely the system is to contain unknown and hidden vulnerabilities. These capabilities can be measured, e.g., by looking at the *experience and education* of evaluators. Quantitative and measurable metrics for human verification include count of verifiers, verifiers' experience in years, number of performed evaluations, as well as the scientific author metrics (number of fresh related publications of the evaluated cryptographic system).

Similarly, other verifiable and relevant public demonstrations of expertise can be included. For example, the number of relevant CVEs (Common vulnerabilities and exposures) [50] submitted by the evaluators should be considered. The evaluators' effectiveness depends on the available software and hardware facilities, as well as on the quality of the processes in the evaluating community or laboratory. Qualitative metrics for effectiveness include *maturity and acceptance of evaluating* laboratory or method by scientific community and organizations. Similarly, qualitative metrics for human effectiveness includes *evaluator's reputation*, which is based on past performance. Effectiveness does not necessarily ensure accuracy. Even though an evaluating person, laboratory, or method detects large amount of known vulnerabilities it may also miss many.

The *openness* of the source code or design (in case of a hardware system) is a particular metric, which affects to the effectiveness of verification. This metric is semi-measurable and qualitative. The possible values are {`open source, closed source, auditor access`}. The last one means that the code is closed to the general public, but is available for auditing by some parties outside the implementors (e.g., officials or potential clients).

Verification time refers to the hours, months, or years that have been spend on exploring the cryptographic solution against vulnerabilities. Time accumulates from intensive product evaluations as well as from the verification and testing by scientific and user community during the system lifetime. The older and more dispersed the system is, the less unknown weaknesses it is likely to have. This is a quantitative and measurable metric.

2) Maturity Metrics: The maturity metrics measure how ready and suitable a cryptosystem is. This can be a metric for a specific component as in Technology Readiness Level (TRL) [52], its derivates such as Systems Readiness Level (SRL) [53] and Integration Readiness Level (IRL) [54], or a more comprehensive metric of a whole systems, such as the Privacy Enhancing Technologies (PETs) maturity metric [55].

TRL measures the readiness of a single component. This metric is measurable and quantitative. *IRL* measures the readiness of components to be integrated to form a more complex system. This metric is measurable and quantitative. *SRL* measures the readiness of a complete system based on the TRLs and IRLs of the different components. The metric is measurable (if normalized) and quantitative. *PETs maturity metric* is a measure for the quality and readiness of privacy enhancing technologies. The PETs measurement is carried out with both measurable indicators (such as the number of papers/patents and lines of code) and a more heuristic

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

evaluation by experts. There is a defined procedure on how to reach consensus on possibly differing evaluations by experts. In some sense, this is similar to the jury evaluation used in some cryptographic standardisation competitions. In the PETs maturity metric, the evaluation is open and transparent, whereas in some cryptographic competitions this is not the case. The PETs maturity metric is measurable and qualitative.

As already mentioned in the introduction, *key length* is one of the most used metrics for cryptosystem security. In our taxonomy, key length considers the maturity and verification level that a cryptosystem has. It is an indicator that shows if the security parameters of a cryptosystem are up to date and provide wanted security against known threats. Key length defines the upper-bound of algorithm security. As different algorithms do not provide equivalent security, estimates on the corresponding strength are made to enable comparisons (e.g., 128 bit symmetric key corresponds to 2000 bit factoring modules and 250 bit elliptic curve keys [14]). The metric is measurable and quantitative.

3) Cost and Performance Metrics: The feasibility of cryptographic products depends not only of their security strength, but also on cost and performance. These metrics can be calculated for the whole system in total or separately for an individual role (e.g., decrypter, encrypter, signer, verifier). Typically, costs are divided unevenly between different roles. This asymmetric cost division may be beneficial, e.g., in cloud or Internet of Things scenarios where another party has more resources available for cryptographic operations. Costs for attackers are estimated with the adversarial resource metrics in Subsection III-A; this subsection focuses to defenders' costs.

Time costs originate from the computations, such as key generation, encryption and decryption, as well as public and private key operations, and from communications, where cryptography causes additional overhead, expands communication and negotiations. Time costs can be estimated by counting elementary operations that a cryptographic solution implies or by experimenting and benchmarking, see e.g., [56]. Typical units for measurement are computing cycles per encrypted block or throughput (bits/second) within particular CPU frequency. Time cost is a quantitative and measurable metric.

Memory and transmission costs relate to the need for run-time and storage memory, as well as to the communication bandwidth. They depend on the sizes of keying material, ciphertexts, and signatures, as well as on run-time memory requirements of algorithms. This is a quantitative and measurable metric, which is typically presented as bit as bytes.

Implementation complexity relates to the size and costs of software or hardware implementations. A key attribute is whether the solution is suitable for standard computing platforms (e.g., Intel x86 or ARM-based) or whether it requires specialized hardware. An important attribute is also whether the performance of the algorithm can be improved with special hardware, such as parallel platforms or extended instruction sets. Complexity can be estimated either by counting lines of code or by counting required hardware resources like gate counts. This is a semi-measurable (as there are many ways to measure complexity) and qualitative metric.

Energy efficiency depends on the use of computing, memory, and communication resources and their costs in different platforms. Hardware computing factors affecting to energy efficiency include, e.g., area size, amount of gates, bit transitions per clock cycle, cycles per algorithm execution, as well as block size [57]. Energy efficiency is a quantitative metric that can measured using joules/bits, watts, or in some cases through the environmental emissions.

IV. Case Studies for Applying Taxonomy

We applied the new taxonomy to evaluate both cryptographic implementations as well as standards. First, we applied the taxonomy to six use cases, where information about a cryptosystem was available through a formal requisition process. All of these systems represent closed-source products, from which all of the technical details may not be available.

Second, we evaluated three different settings and versions from a readily available standard, TLS, with the help of the metric. As a simple test to the metric, we chose to limit our sources to the TLS standards themselves and obvious immediate references such as AES and RSA standards. It can be argued that this is a limiting factor of our analysis. However, it is also important to notice that these metrics should be fairly simple to use and that having a thorough expert analysis through all the relevant research literature on the subject is not possible in most cases, where these metrics should be applied.

A. Measuring Closed-Source Products

We applied both the versions of the taxonomy to six closed-source cryptographic products performing communications security and requiring separate certification before approval for use. Our aim was to see to which extent the metrics could differentiate products and how readily they were available. The results can be seen in Table II.

All of the products aim to provide the same type of security service, and represent the newest versions of their vendors. For this reason, it should be expected that certain baselines and standards will be identical for these products, including their metrics. Furthermore, the metrics were originally designed to cover many types of use cases and algorithms, but in this case study, the products only concentrate on one or two types. Thus, not all metrics can be considered valid for all products.

In addition, due to their closed-source nature, measurement could only be performed with varying degrees of success, as not all information was available, or even asked for in the first place. We have then presented the average availability of the metrics with six levels (1 to 6) as follows:

- Level 1: Metric explicitly stated
- Level 2: Metric implied by an open standard (including independent research on the standard)
- Level 3: Metric implied by a closed standard
- Level 4: Metric implied otherwise
- Level 5: Metric released to evaluators only, via formalized process
- Level 6: Metric withheld

In some cases the metric concerned such features that were not supported, or were not at all available (not even for the manufacturer) or requested in any part of either the procurement or assurance process.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

TABLE II. RESULTS OF TESTING THE METRIC TAXONOMY IN A PROCUREMENT PROCESS

Main category	Subcategory	Metric	Diff.[a]	Avail.[b]	Avg. diff.	Avg. avail.
Adversarial model	Degrees of freedom	Observe/choose/choose adaptively	2	3.3		
		Corruption power - num. of principals	0	6.0		
		Corruption power - degree of corruption	1	3.4		
		Security game compliance	2	3.7		
	Adv.avail. Inform.	Pre-crypto	0	2.8		
		Post-crypto	0	2.8		
		Secret key material	0	2.5		
		Protocol runs	1	3.2		
		Setup parameters	3	3.8		
		Simulation environment	0	N/a		
	Adversarial goal	Goal	1	3.3		
	Adv. resources	Computation power Instantiated	0	4.5		
		Computation power Non-instantiated	0	4.0		
		Memory Instantiated	0	4.5		
		Memory Non-instantiated	0	4.0	0.7	3.7
Proof framework	Sec. assumptions	Mathematical complexity	0	3.0		
	Abstr. assumptions	Type	0	3.0		
		Num. of assumptions	0	3.0		
		Maturity of assumptions	0	3.0		
	Methodology	Tightness	0	3.0		
	Rigor	Rigor	0	3.0	0.0	3.0
Verif.&feas.	Assurance levels	Assurance standard or profile	6	1.0		
		Evaluation assurance level	6	1.0		
	Test coverage	Percentage of covered areas	5	5.2		
	Vulnerabilities	Number of detected vulnerabilities	2	2.7		
	Side-channels	Existence	3	5.3		
	Metadata	Leaked amount and type	4	2.8		
		Openness	3	1.0		
	Human efficiency	Academic publications	3	5.4		
		Verifier experience in years	3	5.4		
	Verification time	Time since released for evaluation	4	2.2		
		Size and efforts of eval. community	5	4.5		
	Readiness level	Technology readiness level	0	-		
		Integration readiness level	0	-		
		System readiness level	0	-		
		PETS maturity model	0	-		
	Key length	Bits for criteria compliance	6	1.8		
	Time costs	Execution overhead	6	1.0		
		Communication overhead	5	3.7		
	Mem. & BW	Run-time memory	5	5.2		
		Storage capacity	5	5.2		
		Communication bandwidth	6	1.0		
	Impl. compl.	Size of software	5	5.2		
		Dedicated hardware requirements	3	1.5		
	Energy efficiency	Algorithm complexity dependent joules	0	6.0		
		Hardware platform dependent joules	6	1.0	3.6	3.2

[a] How many products this metric can differentiate.[b] Availability of the metric (average of the values for the availability levels as specified in Section IV-A).

The first version of the taxonomy proved to be too vaguely described in the adversarial model and security framework parts, and could not be readily applied to product level. Thus we applied the taxonomy fully only to the current revised version. The three categories did not, surprisingly, have significant differences in the availability of the metrics on behalf of the vendors. This may be partly due to standardization, but also due to the fact that many performance and security evaluation details are actually company secrets. The algorithmic metrics were slightly more difficult to obtain, due to some closed standards prevalent in the field.

The second "metric of a metric" we measured was how many products (of the six) a particular metric was able to differentiate (technically "five" is redundant, since the sixth could be identified by being the remaining one, but we consider the applicability only).

According to this differentiation ability, there is a striking difference between performance and verification metrics compared to others:

- None of the proof framework metrics could differentiate between products (either due to standardization or to complete unavailability)
- The adversarial model metrics could make a difference between products only on protocol level (i.e., key exchange) details, not primitives (like block cipher) themselves.
- The verification and feasibility metrics, however, were able to distinguish 3.6 or closer to 4 products out of 6. This may reflect the very strict control on security features and a stable set of use cases, forcing the vendors to compete on the performance rather than security.

Some notes for individual metrics include:

- The assumption for non-instantiated computer power and memory for the adversary are implicitly "PPT"

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

(/"PSPACE"), unless quantum security is considered. At this point, no off-the-shelf product we looked at offered quantum-secure constructions, thus these metrics would not be expected to provide any difference.
- If some metric is promoted by authorities and standards, it will usually eventually become part of public sales brochures or public knowledge in any case. One good example of this is the cryptographic key length: it is probably one of the most exact and widespread metrics in cryptography, required to be specified very ubiquitously. This has led even closed standard algorithms to divulge or leak their key lengths (e.g., BATON [58], and Libelle [59]). This is also evidenced by our study: key length is one of the few metrics we could pinpoint in all of the test subjects.

The adversarial model metrics mostly concern algorithm level issues, which are common to standardized products. The models are developed independently and publicly inside the cryptologic community, and although they may not be part of a standard, they are implicitly understood to follow from cryptographic research. Many algorithmic metrics of standards may be difficult to locate in the cryptographic literature for a non-expert in cryptographic theory. Thus it is advisable that the most important standards be readily measured in a table (for example) by a group of experts beforehand. Examples include the UC-security of (a portion of a version of) TLS [60] and IND-CCA security of IPSec IKEv2 [61]. Sometimes it is not easy to see, whether a metric actually is included in another, e.g., whether UC-security includes adaptive adversaries [62].

The main outcome of this case study is that algorithmic best practices are usually incorporated into various standards, which many products will follow. Thus, if the standards used are very uniform, such as the case with AES, algorithmic metrics are unlikely to make a difference between products. However, in cases where a standard allows multiple solutions (such as the planned PQC-standard), there are no standards (many closed-market products and new innovations such as blockchains and homomorphic encryption) or the standards are vague (e.g., implementation details, parametrization, protocol-level solutions), all the metrics are likely to serve a definite purpose also from end-user perspective. This case study was not able to weigh the proposed metrics across product types or across standards.

B. Measuring TLS 1.2 and TLS 1.3

For TLS 1.2 [63] we evaluated one setting, which was the ciphersuite `TLS_RSA_WITH_AES_128_CBC_SHA`. All TLS 1.2 implementations must support this ciphersuite according to the standard.

We used the taxonomy in [1] and searched through the TLS 1.2 standard for valuations of the different metrics in the taxonomy. For many metrics the standard did not contain suitable answers. Then also the references in the standard were studied. In several cases we also needed to do larger searches for example to find out about the number of vulnerabilities.

Finding the correct framing for measuring all the metrics was not an easy task. In some cases the valuation was left empty or Not applicable was used. For TLS this was the case for the different readiness levels as there is no authoritative source for this type of information.

For TLS 1.3 [64] we measured two settings: `TLS_AES_128_CCM_SHA256` and `TLS_AES_128_CCM_SHA256`. The former was chosen as the one closest to the TLS 1.2 ciphersuite `TLS_RSA_WITH_AES_128_CBC_SHA`, although the differences between the two versions are so vast that a perfect match does not exist. The latter was the mandatory setting for TLS 1.3.

The first thing we noticed when applying the metric to the TLS cases was that the adversarial model metrics are essentially the same for all three cases, since they are about the freedom and powers granted to the adversary when constructing the proofs for the underlying cryptographic primitives. The documentation we were using did not provide direct answers to these metrics, instead prior knowledge, or expert opinion, needed to be used (or further research if we were not artificially restricting our sources). For example, we could say that the adversary would have "substantial resources" to get to the goal of "information leak", but that would not come from the TLS documentation, or any of the other specifications we were using.

Differences between standard versions are in the performance and in the security and feasibility of implementations. Metrics related to the performance and new metrics related to the side-channel and metadata can capture these differences. Further, TLS 1.3 specification mitigates some previous attacks that were related, e.g., to renegotiation, protocol version downgrading, and compression [64]. These advances are visible in the verification metric category, where number of known vulnerabilities are counted and indirectly also in the readiness level metric where knowledge of existing vulnerabilities should be visible as the lowered maturity evaluations. When considering the evaluation efforts, both protocols have passed similar large scale reviews by global security community and industry. TLS 1.2 may however seen more evaluations as it was released for evaluation 2008, about ten years before release of TLS 1.3. There are also some formal verification efforts related to TLS, e.g,, in [65].

V. Discussion

Measuring the security of systems is a very difficult task even though the area has been researched for a long time and the interest has been increasing in recent years. Measuring the security and strength of cryptosystems seems to be even harder, because there are so many different metrics and variables involved and these also interact in many ways. Furthermore, the notion of security is very much context dependent. Even a secure cryptographic primitive used in a wrong context provides very little security. An insecure primitive in a wrong context provides even less security, e.g., [66].

The reason for revising the earlier taxonomy of [1] was the difficulty in using that taxonomy in gaining actionable information of real cryptosystems. Our new taxonomy should alleviate this situation, but there is still room for improvement and future work. We have focused more on metrics, which are relevant for the development and operational phases in the life cycle of cryptographic systems. We have given more concrete

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

metrics for addressing implementation specific threats, side-channels and leaked metadata.

As stated before, correctness, measurability and meaningfulness are core quality criteria for security metrics. We investigated the metrics in use cases, giving enough contextual specificity and same time dependability, both contributing to correctness. Two other dimensions of correctness, granularity and completeness are difficult to be analyzed based on the use cases, and would require more experimentation. Most of the metrics presented in the revised taxonomy are measurable, or at least semi-measurable. Investigating meaningfulness of the taxonomy's metrics would also require more extensive studies. However, clarity and applicability to decision-making were driving the taxonomy development.

One key issue already pointed out in [1] is that of dependencies between different metrics. For instance, feasibility metrics, costs and performance, depend on the algorithmic model. The revised taxonomy is still open for new inter-metric derivatives. For instance, there are some measures that we have chosen, due to simplicity, not to present in our taxonomy as their own separate metrics. One is the total (monetary) cost of an attack as a resource metric. It could be argued that this is one of the most relevant metrics there is. On the other hand, it is also derivative of the adversarial resource metrics that have been included in our taxonomy. A major direction of future improvement of these metrics will be to quantify the relationships as well as to find more independent measures. With independent metrics, it could be easier to produce information on cryptosystems that is understandable and also usable to the different stakeholders measuring cryptosystems.

As mentioned already in the beginning of this paper, the utility of any metric is in its applicability to real use cases and the ability to discern between different cryptosystems. Thus, the new taxonomy needs to be applied in some use case to determine its usefulness. Then also new improvements can be made.

It is also important to note, that for example in the verification metrics category there are many standards that can be applied to cryptosystem evaluation. However, having for example good coverage of a given testing standard, e.g., FIPS 140-2 [67], does not necessarily mean that the randomness generation can withstand adversarial attacks [68]. Furthermore, there are cases such as the Dual-EC, where a backdoor was included in the standard and remained there for quite some time [69]. Thus, even good values in this type of coverage metric can not guarantee good security without other information or further metrics.

One clear shortcoming of both the original metrics from [1] and our revised version is the results of many metrics being incomparable. This means that it is hard to get an absolute ordering of cryptosystems and their security. With truly independent metrics that are also all quantitatively measurable, one could compute (a possibly weighed) average of the different values and use that as the score for the system. These could then be ordered easily. The current ensemble of different metrics gives only a vector in a space, where distances are not well defined or do not even exist in any meaningful way.

VI. Conclusion

In this paper, we have revised the taxonomy of metrics for cryptographic systems from [1] to a new taxonomy. We have used this new taxonomy to evaluate both closed source implementations and the open TLS 1.2 and TLS 1.3 standards. Although there are some metrics where differentiation can be achieved, the taxonomy and the different metrics in it do not provide yet a usable tool for evaluations. The process of evaluating the metrics and differentiating between different implementations is fairly complex and the results are not necessarily decisive.

For future work, there is still a great need to improve the metrics and to find ways to simplify the measurement process. Whether this can be achieved through revision of the taxonomy presented here or through some different approach, is an important venue for further study.

References

[1] K. Halunen, J. Suomalainen, O.-M. Latvala, M. Kylänpää, V. Vallivaara, and M. Kiviharju, "A taxonomy of metrics for cryptographic systems," in Thirteenth International Conference on Emerging Security Information, Systems and Technologies, SECURWARE, 2019.

[2] P. C. van Oorschot, "Public key cryptography's impact on society: How diffie and hellman changed the world."

[3] Z. Benenson, U. Kühn, and S. Lucks, "Cryptographic Attack Metrics," in Dependability Metrics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 133–156.

[4] N. Jorstad and T. S. Landgrave, "Cryptographic algorithm metrics," 20th National Information Systems Security, 1997, http://csrc.nist.gov/nissc/1997/proceedings/128.pdf, (accessed 9.12.2020).

[5] EN ISO/IEC 19790:2020, "Information technology. Security techniques. Security requirements for cryptographic modules (ISO/IEC 19790:2012)," International Organization for Standardization, Geneva, CH, Standard, 2020.

[6] ISO/IEC 29128:2011(E), "Information technology — Security techniques — Verification of cryptographic protocols," International Organization for Standardization, Geneva, CH, Standard, 2011.

[7] NIST, "Cryptographic Algorithm Validation Program," https://csrc.nist.gov/Projects/Cryptographic-Algorithm-Validation-Program, (accessed 9.12.2020), 2018.

[8] PCI Security Standards Council, "Payment card industry data security standard v3.2.1," 2018, https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2-1.pdf, (accessed 9.12.2020).

[9] International Telecommunication Union, "Global Cybersecurity Index (GCI) 2018," 2018.

[10] P. Emami-Naeini, H. Dixon, Y. Agarwal, and L. F. Cranor, "Exploring how privacy and security factor into iot device purchase behavior," in Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019, pp. 1–12.

[11] R. Savola, C. Frühwirth, and A. Pietikäinen, "Risk-driven security metrics in agile software development: An industrial pilot study," Universal Computer Science, vol. 18, 2012, pp. 1679–1702.

[12] R. M. Savola, "Quality of security metrics and measurements," Computers & Security, vol. 37, 2013, pp. 78–90.

[13] N. Smart, "Algorithms, Key Size and Protocols Report (2018)," Tech. Rep., 2018.

[14] BSI, "Cryptographic mechanisms: Recommendations and key lengths," https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TG02102/BSI-TR-02102-1.pdf, (accessed 9.12.2020), BSI TR-02102-1, Tech. Rep., 2018.

[15] M. Carvalho, J. DeMott, R. Ford, and D. A. Wheeler, "Heartbleed 101," IEEE Security Privacy, vol. 12, no. 4, 2014, pp. 63–67.

[16] A. Biryukov and D. Khovratovich, "Related-key cryptanalysis of the full aes-192 and aes-256," in International Conference on the Theory and Application of Cryptology and Information Security. Springer, 2009, pp. 1–18.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[17] A. Joux, "Multicollisions in iterated hash functions. application to cascaded constructions," in Annual International Cryptology Conference. Springer, 2004, pp. 306–316.

[18] J. Kortelainen, K. Halunen, and T. Kortelainen, "Multicollision attacks and generalized iterated hash functions," Journal of Mathematical Cryptology, vol. 4, no. 3, 2010, pp. 239–270.

[19] M. A. AlAhmad, I. F. Alshaikhli, and M. Nandi, "Joux multicollisions attack in sponge construction," in Proceedings of the 6th International Conference on Security of Information and Networks, ser. SIN '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 292–296. [Online]. Available: https://doi.org/10.1145/2523514.2523551

[20] Common Criteria, "Common Criteria for Information Technology Security Evaluation - Part 1: Introduction and general model," Common Criteria, vol. 3.1, no. April, 2017, pp. 1–106.

[21] ISO/IEC 24759:2017(E), "Information technology — Security techniques — Test requirements for cryptographic modules," International Organization for Standardization, Geneva, CH, Standard, 2017.

[22] Microsoft, "Bitlocker," https://docs.microsoft.com/en-us/windows/security/information-protection/bitlocker/bitlocker-overview, (accessed 9.12.2020), 2018.

[23] M. Bellare and P. Rogaway, "Optimal asymmetric encryption," in Advances in Cryptology—EUROCRYPT'94. Springer, 1995, pp. 92–111.

[24] S. Goldwasser, S. Micali, and R. L. Rivest, "A digital signature scheme secure against adaptive chosen-message attacks," SIAM Journal on Computing, vol. 17, no. 2, 1988, pp. 281–308.

[25] A. W. Cross, L. S. Bishop, S. Sheldon, P. D. Nation, and J. M. Gambetta, "Validating quantum computers using randomized model circuits," Phys. Rev. A, vol. 100, Sep 2019, p. 032328. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.100.032328

[26] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, and Y. Markov, "The first collision for full SHA-1," in Annual International Cryptology Conference. Springer, 2017, pp. 570–596.

[27] L. Valenta, S. Cohney, A. Liao, J. Fried, S. Bodduluri, and N. Heninger, "Factoring as a service," in International Conference on Financial Cryptography and Data Security. Springer, 2016, pp. 321–338.

[28] J. Sugier, "Cracking the DES cipher with cost-optimized FPGA devices," in International Conference on Dependability and Complex Systems. Springer, 2019, pp. 478–487.

[29] R. Canetti and T. Rabin, "Universal composition with joint state," in Annual International Cryptology Conference. Springer, 2003, pp. 265–281.

[30] R. Canetti, Y. Dodis, R. Pass, and S. Walfish, "Universally composable security with global setup," in Theory of Cryptography Conference. Springer, 2007, pp. 61–85.

[31] A. Datta, R. Küsters, J. C. Mitchell, and A. Ramanathan, "On the relationships between notions of simulation-based security," in Theory of Cryptography Conference. Springer, 2005, pp. 476–494.

[32] K. Auguste, "La cryptographie militaire," Journal des sciences militaires, vol. 9, no. 538, 1883, p. 5.

[33] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," in Proceedings 2001 IEEE International Conference on Cluster Computing. IEEE, 2001, pp. 136–145.

[34] R. Kusters, "Simulation-based security with inexhaustible interactive turing machines," in 19th IEEE Computer Security Foundations Workshop (CSFW'06). IEEE, 2006, pp. 12–pp.

[35] M. Backes, B. Pfitzmann, and M. Waidner, "Limits of the reactive simulatability/UC of Dolev-Yao models with hashes," 2006.

[36] M. Prabhakaran and A. Sahai, New notions of security. Princeton University, 2005.

[37] D. Dolev and A. Yao, "On the security of public key protocols," IEEE Transactions on information theory, vol. 29, no. 2, 1983, pp. 198–208.

[38] P. Syverson, C. Meadows, and I. Cervesato, "Dolev-Yao is no better than Machiavelli," Naval Research Lab, Washington DC, USA, Center for high assurance computing systems, Tech. Rep., 2000.

[39] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in Proceedings of the 1st ACM conference on Computer and communications security. ACM, 1993, pp. 62–73.

[40] P. Ananth and R. Bhaskar, "Non observability in the random oracle model," in International Conference on Provable Security. Springer, 2013, pp. 86–103.

[41] J. B. Nielsen, "Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case," in Annual International Cryptology Conference. Springer, 2002, pp. 111–126.

[42] V. Shoup, "Lower bounds for discrete logarithms and related problems," in International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 1997, pp. 256–266.

[43] C. E. Shannon, "Communication theory of secrecy systems," Bell system technical journal, vol. 28, no. 4, 1949, pp. 656–715.

[44] R. Canetti and M. Fischlin, "Universally composable commitments," in Annual International Cryptology Conference. Springer, 2001, pp. 19–40.

[45] J.-S. Coron, T. Holenstein, R. Künzler, J. Patarin, Y. Seurin, and S. Tessaro, "How to build an ideal cipher: the indifferentiability of the feistel construction," Journal of cryptology, vol. 29, no. 1, 2016, pp. 61–114.

[46] Common Criteria, "Common Criteria for Information Technology Security Evaluation Part 2 : Security functional components," Security, no. April, 2017, pp. 1–323.

[47] NIST, "Derived Test Requirements for FIPS PUB 140-2, Security Requirements for Cryptographic Modules," 2011.

[48] International Organization for Standarization, "ISO/IEC 29128:2011 - Information technology – Security techniques – Verification of cryptographic protocols," 2011.

[49] NIST, "Special publication 800-22. a statistical test suite for random and pseudorandom number generators for cryptographic applications," 2010.

[50] MITRE, "Common vulnerabilities and exposures," https://cve.mitre.org, (accessed 9.12.2020), 1999.

[51] FIRST, "Common vulnerability scoring system," https://www.first.org/cvss/, (accessed 9.12.2020), 2015.

[52] J. C. Mankins, "Technology readiness levels," White Paper, NASA, 1995.

[53] B. Sauser, D. Verma, J. Ramirez-Marquez, and R. Gove, "From TRL to SRL: The concept of systems readiness levels," in Conference on Systems Engineering Research, Los Angeles, CA, 2006, pp. 1–10.

[54] B. Sauser, R. Gove, E. Forbes, and J. E. Ramirez-Marquez, "Integration maturity metrics: Development of an integration readiness level," Information Knowledge Systems Management, vol. 9, no. 1, 2010, pp. 17–46.

[55] M. Hansen, J. Hoepman, M. Jensen, and S. Schiffner, "Readiness analysis for the adoption and evolution of privacy enhancing technologies: Methodology, pilot assessment, and continuity plan," Tech. rep., ENISA, Tech. Rep., 2015.

[56] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, "A survey of lightweight-cryptography implementations," IEEE Design & Test of Computers, vol. 24, no. 6, 2007, pp. 522–533.

[57] S. Kerckhof, F. Durvaux, C. Hocquet, D. Bol, and F.-X. Standaert, "Towards green cryptography: a comparison of lightweight ciphers from the energy viewpoint," in International Workshop on Cryptographic Hardware and Embedded Systems. Springer, 2012, pp. 390–407.

[58] Wikipedia, "BATON," en.wikipedia.org/wiki/BATON, (accessed 9.12.2020).

[59] ——, "Libelle (kryptographie)," de.wikipedia.org/wiki/Libelle_(Kryptographie), (accessed 9.12.2020).

[60] S. Gajek, M. Manulis, O. Pereira, A.-R. Sadeghi, and J. Schwenk, "Universally composable security analysis of tls—secure sessions with handshake and record layer protocols," Cryptology ePrint Archive, Report 2008/251, 2008, https://eprint.iacr.org/2008/251, (accessed 9.12.2020).

[61] A. Roy, A. Datta, and J. C. Mitchell, "Formal proofs of cryptographic security of diffie-hellman-based protocols," in International Symposium on Trustworthy Global Computing. Springer, 2007, pp. 312–329.

[62] D. Hofheinz, J. Mueller-Quade, and R. Steinwandt, "Initiator-resilient universally composable key exchange," Cryptology ePrint Archive, Report 2003/063, 2003, https://eprint.iacr.org/2003/063.

[63] T. Dierks and E. Rescorla, "The transport layer security (TLS) protocol version 1.2," 2008.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[64] E. Rescorla and T. Dierks, "The transport layer security (TLS) protocol version 1.3," 2018.

[65] C. Cremers, M. Horvat, J. Hoyland, S. Scott, and T. van der Merwe, "A comprehensive symbolic analysis of tls 1.3," in Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1773–1788. [Online]. Available: https://doi.org/10.1145/3133956.3134063

[66] F. Y. Rashid, "Adobe's hacked passwords: They are terrible!" https://uk.pcmag.com/opinion/12060/adobes-hacked-passwords-they-are-terrible, November 2013.

[67] NIST, "Security Requirements for Cryptographic Modules. FIPS PUB 140-2," 2001.

[68] D. Hurley-Smith, C. Patsakis, and J. Hernandez-Castro, "On the unbearable lightness of fips 140-2 randomness tests," IEEE Transactions on Information Forensics and Security, 2020.

[69] D. J. Bernstein, T. Lange, and R. Niederhagen, "Dual ec: A standardized back door," in The New Codebreakers. Springer, 2016, pp. 256–281.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

A Domain-agnostic Framework for Secure Design and Validation of CPS Systems

Rohith Yanambaka Venkata, Nathaniel Brown, Rohan Maheshwari and Krishna Kavi

Dept. of Computer Science and Engineering
University of North Texas
Denton, Texas-76207
Email: {nathanielbrown, rohanmaheshwari}@my.unt.edu {ry0080, krishna.kavi}@unt.edu

***Abstract*—Cyber Physical Systems (CPS) are an integration of computational and physical processes, where the cyber components monitor and control physical processes. Cyber attacks largely target the cyber components with the intention of disrupting the functionality of the components in the physical domain. In this paper, we present SIMON, an Ontological design and verification framework that captures the intricate relationship(s) between cyber and physical components in CPS by leveraging standard Ontologies and extending the NIST CPS framework for the purpose of eliciting trustworthy requirements, assigning responsibilities and roles to CPS functionalities and validating that the trustworthy requirements are met by the designed system. We demonstrate the capabilities of SIMON using a vehicle to infrastructure (V2I) safety application. In addition, we also investigate introducing resiliency measures that will ensure compliance of physical systems to specifications.**

Keywords–CPS Security; Ontology; CPS Privacy; CPS Resiliency; Semantic Web

I. INTRODUCTION

CPS systems consist of electronic or computer systems that control physical systems. These systems use sensors to collect information about the physical system and possibly other situational inputs, process these inputs to determine appropriate decisions and affect these decisions on the physical system via actuators, forming a consistent feedback loop between the physical and computational realms. Additionally, the data collection and transmission of actions may involve the use of communication networks. Well-known applications of CPS systems include smart automobiles, manufacturing including additive (3-D) manufacturing, medical monitoring equipment and smart grids.

The increased reliance on such CPS systems in everyday life has resulted in a corresponding increase in available venues for attack for malicious actors. In contrast with information security, which primarily deals with the protection of valuable information, CPS systems offer attackers the potential to affect the physical world through digital means. Thus, it is essential to understand the inter-relationships between the functions of the physical systems and the cyber (or electronic) systems, and how an attack on one affects the other. In this paper, we present an extension of our prior work on a design validation framework that will enable the design of secure CPS systems [1].

Since CPS systems can contain sensors, actuators, electronic/processing components and communication networks, the number of sources receiving and transmitting information is large when compared to traditional systems that fall under a more strict cyber or physical definition, providing many opportunities to attackers who want to impact the digital or physical realm, or both. Real attacks have been carried out on both power grids and interconnected industrial control systems (ICS); such large-scale attacks are predominantly carried out on the "nation or state actors" [2]. Potential attacks include the purposeful disablement or modification of connected medical equipment vital to patient survival, disablement of smart car brakes leading to collisions, and the sabotage of industrial processes to bring harm to industrial production cycles and/or human workers [2][3]. The increased number of demonstrated and theoretical attacks has prompted a response from the cybersecurity community to attempt to develop frameworks and models to address CPS system security concerns [4].

A primary challenge of conceptualizing CPS system security is determining which threats and corresponding security recommendations apply to CPS systems in general, and which are unique to a specific domain. Another challenge is managing the complexity that arises from CPS systems' dual nature of participating in both the cyber and physical realms, referred to as its heterogeneity. Humayed et al. [2] emphasize how CPS systems should satisfy the three traditional information security requirements—confidentiality, integrity, and availability—as well as safety, a fourth metric specific to the physical nature of CPS sytems. Ashibani and Mahmoud [5] recommend a security analysis at the perception, transmission, and application layers of CPS systems. Such static analyses are helpful for beginning to diagnose vulnerabilities in CPS systems and address them through actionable steps. However, the interconnected nature of CPS systems leaves a desire for a modeling framework that can account for the high complexity of CPS systems and the tendency toward human error.

To address these concerns, we advocate the use of Ontologies to model CPS systems and the relationships between their constituent subsystems. An Ontology is a formal description of knowledge as a set of concepts within a domain and the relationships that hold between them [6]. To enable such a description, we need to formally specify individuals (instances of objects), classes, attributes, and relations as well as restrictions, rules, and axioms. Ontologies not only enable a shareable and reusable knowledge representation but, can also add new knowledge about a domain [6]. Further, we extend the NIST CPS framework [7], which includes 3 phases: conceptualization for capturing requirements of the systems, realization, which describes the design and implementation, and assurance, which enables verification of requirements. In our SIMON framework, we subdivide realization phase by differentiating between an abstract realization and a concrete realization levels. The abstract level translates the conceptual requirements of CPS systems (such as functional, timing, trustworthiness requirements) into responsibilities and roles of system components (such as sensors, actuators, processing elements, communication systems, computational algorithms). The concrete realization level defines specific products used to implement the abstract responsibilities and functionalities (such as selecting a specific IoT system, or a communication

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

device). Our Ontologies allow for common vocabularies to describe concepts and properties of CPS systems at various levels of the design framework. This permits for adapting best design practices of one domain to the design of systems in another domain.

Our prior work on using Ontologies in vulnerability assessment in cloud systems [8] [9] enables us to extend those Ontologies to address security concerns in CPS systems. Using the NIST CPS framework as a basis for SIMON allows for a broad and integrated view of CPS and positions trustworthiness, among other aspects of CPS design. Furthermore, using standard Ontologies like SOSA [10] will help streamline the process of secure CPS design by considering the properties of a CPS system like sensing and actuation.

The rest of the paper is organized as follows. Section III describes SIMON, our proposed CPS framework. This section also describes the various standard Ontologies, as well as some of our new Ontologies used in the framework. Section IV includes two case studies to show how SIMON can be used for the design and validation of CPS systems. We show some examples of cyber attacks and use reasoners to identify potential compromise of design goals associated with the physical system.

II. Related Work

Extensive research has been conducted in applying Ontologies to either identify or validate the security posture of CPS or IoT systems. Mozzaquatro et al. [11] propose a framework that employs a model-driven approach to designing secure CPS systems. While this may be prudential in some domains, it fails to account for concerns from various stakeholders in a CPS system. This is addressed by the NIST CPS framework [7].

Fenz et al. [12] and Settas et al. [13] propose Ontological frameworks that are complemented by Bayesian analysis to predict threat probabilities in cloud systems. The key competencies of these contributions is vulnerability assessment and threat modeling for cyber systems in the cloud. These frameworks are not directly applicable to CPS systems because they do not account for the physical components of these systems. Moreover, additional vulnerabilities exist in the intersection between cyber and physical components in a CPS system. Modeling this interaction is essential in understanding the impact of a potential compromise.

Gonzalez-Gil et al. [14] describe an Ontology for Machine to Machine (M2M) data security in Internet of Things (IoT) systems. The focus of this work is to define a semantic framework to facilitate knowledge sharing and improve security of IoT systems. The Ontology describes various data security traits involved in data access and exchange in IoT systems. Its purpose is to serve as a common vocabulary supporting the description of the security mechanisms associated with data and data exchange, which are strategic and crucial in varied domains, such as data provisioning, service aggregation and data processing [14]. While the knowledge sharing property of an Ontology is leveraged in this work, the logical reasoning property is not. Hence, the true capability of Ontologies is not fully harnessed.

Bhandari and Gujral [15] present a semantic approach to modeling the security posture of a network. A computer network is a dynamic entity with a constantly changing topology. The addition and removal of new services, hardware components and sub networks, and modification of new user roles contribute to the dynamic status of a network. This work can be considered a precursor to the Structured Threat Information Expression (STIX) Ontology that is discussed in Section III.

Lannacone et al. [16] describe an Ontology developed from a database of cyber security knowledge graphs. It is intended to provide an organized framework that incorporates information from a variety of structured and unstructured data sources.

Current research investigating the feasibility of semantic technology for security in CPS appears to the limited to knowledge reuse. Several semantic frameworks have been developed to understand the security posture of cyber systems. While these frameworks may provide an insight into the security issues that plague various CPS systems, this information may be unreliable because the frameworks do not account for the tight coupling between cyber and physical components. Furthermore, identifying security concerns at the design stage of CPS systems using Ontologies has not been explored.

SIMON aims to bridge the gap between system design and validation using cyber threat data from multiple sources. We believe that this approach will help in the design of secure CPS systems.

III. SIMON Framework

The proposed framework combines (and extends) existing standard specification Ontologies such as Semantic Sensor Networks (SSN), and develop new ones as required by the domain of interest. Let us take a closer look at some of the Ontologies and frameworks used in our research.

A. NIST CPS Framework

National Institute of Standards and Technology (NIST) has developed a framework that provides guidance in designing, building and verifying complex CPS systems [7]. The framework captures generic functionalities that CPS provide, the activities and artifacts needed to support conceptualization, and realization and assurance of CPS design [7]. Designing a CPS system involves:

- **Conceptualization -** This involves capturing all activities related to high-level goals, functional requirements and organization of CPS as they pertain to what the CPS is supposed to do. It provides a conceptual model of the CPS system under consideration and can be used to capture requirements from different perspectives (such as functional, timing, trustworthiness, business).
- **Realization -** This involves capturing all activities surrounding the detailed engineering, design, production, implementation and operation of the desired systems. However, to facilitate comparing Ontological models of CPS systems, we propose bifurcating the overarching realization phase described in the NIST CPS framework into the following sub-phases.
  - **Abstract Realization -** In this phase, design goals are broken down into roles and responsibilities and delegated to subsystems and interfaces. For example, we may identify that the network communications needed in the system will be handled by a wireless data

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

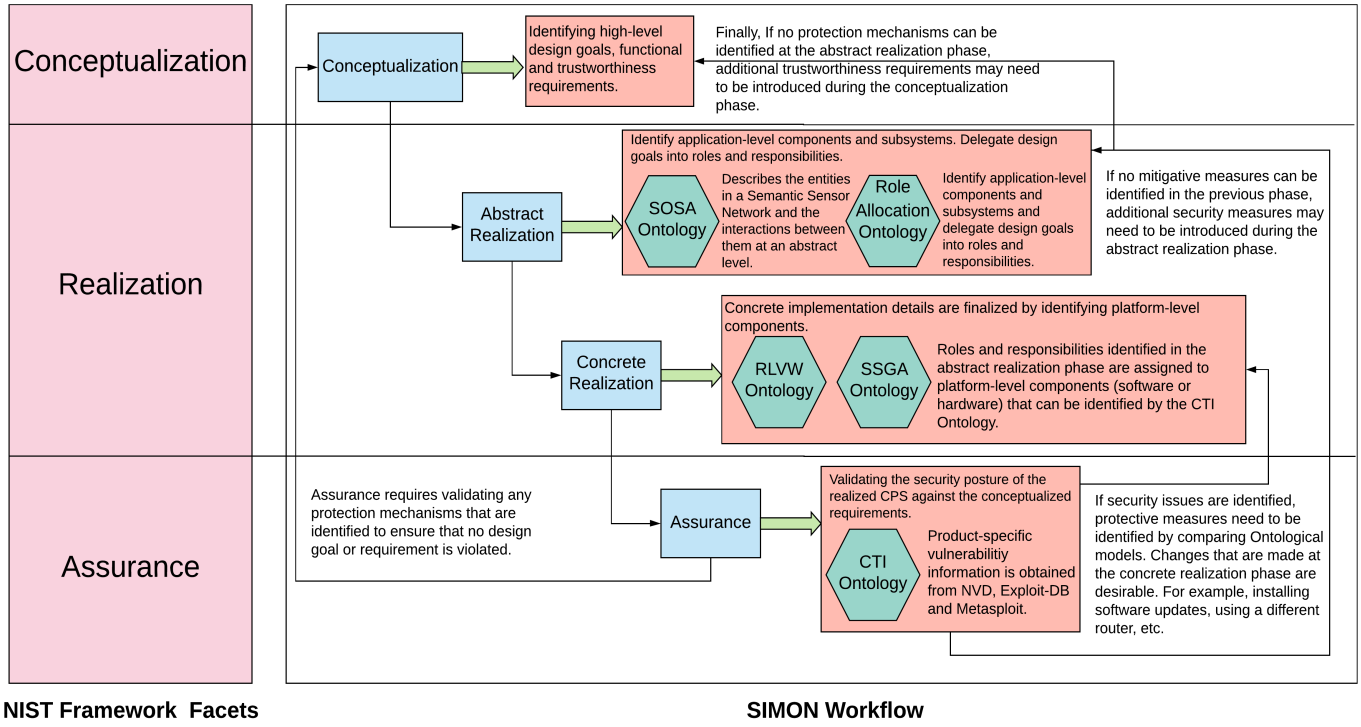


Figure 1. The SIMON Ontological Framework.

communication application but not provide details on either the specific hardware device or communication protocols. We use Ontologies to capture the Abstract Realization.

- **Concrete Realization -** The roles and responsibilities identified during the abstract realization phase need to be implemented by specific products. For example, a Cisco ASR1002-10G-HA/K9 may be selected as the wireless data communication role identified in the Abstract Realization phase. We use Ontologies to relate the products used for various functions and roles identified in the Abstract Realization.

- **Assurance -** The assurance phase deals with obtaining confidence that the system built in the realization phase satisfies the model developed in the conceptualization phase [7]. In our case, we use reasoners to infer and derive assurances (or violations) that the security goals are met. We use additional Ontologies to capture cyber threat data so that vulnerabilities, cyber attacks and possible mitigations can be related to the products identified in Concrete Realization; we rely on NIST Common Platform Enumeration (CPE) identities with specific products for this purpose.

SIMON can be used to modify the CPS design at any of the various phases to address any design violations discovered by our reasoners. Figure 1 describes an abstract view of our framework for the design and verification of CPS systems, focusing on security and trustworthiness.

B. Role Allocation

Requirements traceability is an essential property in identifying changes/modifications to components that will improve the security posture of a CPS system. Delegating design goals from the conceptualization phase into roles and responsibilities for entities identified in the two realization phases will help achieve traceability.

The abstract realization phase involves identifying application-level components, sans the implementation details. Each system identified in this phase can be used to define a role that associates a set of conceptualized functional requirements for the underlying sub-systems to realize. In addition, each role be assigned security responsibilities to be fulfilled. The responsibilities from abstract realization are mapped to the specific concrete realizations: several abstract roles may be assigned to a single concrete component. A detailed example is presented in Section IV.

The trustworthiness requirements as described by the NIST CPS Framework include:

- **Privacy:** Addresses concerns pertaining to the prevention of unauthorized agents gaining access to data stored in, created by or transiting through a CPS system or its components [7].
- **Reliability:** Addresses concerns related to the ability of a CPS to deliver stable and predictable performance in the expected conditions [7].
- **Resilience:** Addresses concerns related to the ability of a CPS to withstand instability, unexpected conditions, and gracefully return to predictable, but possibly degraded performance [7].
- **Security:** Addresses concerns related to the ability of the CPS to ensure that all of its processes, mechanism (both cyber and physical), and services are afforded internal or external protection from unintended and unauthorized access, change, damage, destruction, or use [7]. Security

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

can best be described through three lenses:

- **Confidentiality:** Preserving authorized restrictions on access and disclosure.
- **Integrity:** Guarding against improper modification or destruction of system, and includes ensuring non-repudiation and authenticity
- **Availability:** Ensuring timely and reliable authorized access to and use of a system.

We use several different Ontologies in our framework to describe the concepts, properties and restrictions associated with CPS systems at each of the design phases described in this section.

C. Sensor-Observation-Sampling-Actuator Ontology (SOSA)

The Sensor-Observation-Sampling-Actuation Ontology (SOSA) [10], a subset of the Semantic Sensor Network (SSN) Ontology, presents a conceptualization of all entities, activities and properties that typically constitute a CPS. SOSA is a World Wide Web Consortium (W3C) standard specification.

The *core structure* of SOSA Ontology encompasses all of the three modeling perspectives of sensors and actuators; the activities of observing, sampling, and actuating [10]. Each activity targets a feature of interest by either changing its state or revealing its properties by following a designated procedure. All activities are carried out by an object, also called an *agent*.

SOSA aims to strike a balance between the expressivity of the underlying description logic, the ease of use of language features and the expectations of the target audience, while accommodating a broad range of domains and applications [10].

D. Cyber Threat Information Ontology

The activities of observing and sampling must be followed by communicating and processing the data to interpret the observations and making decisions on the actions. These actions are then used to control physical systems through actuation. The communication and processing subsystem, which is not directly included in the SOSA Ontology, can expose the cyber and physical components of the CPS to security attacks. Thus, SOSA must be extended to describe the processing and communication subsystems. This allows us to relate cyber threat data from multiple sources to obtain insights into the security posture of a CPS system under consideration. We have defined an Ontology that captures Cyber Threat Information (CTI) from three sources:

- **The National Vulnerability Database (NVD) -** A U.S. government repository of standards-based vulnerability management data [17].
- **Exploit Database -** An archive of public exploits and corresponding vulnerable software, developed for use by penetration testers and vulnerability researchers [18].
- **Metasploit -** A framework for developing, testing and executing software exploits [19].

Our Ontology can easily be extended to capture CTI from other sources. The cyber threat Ontology is underpinned by the STIX structured language [20], that enables organizations to share, store and analyze CTI in a consistent manner, allowing security communities to better understand what computer-based attacks they are most likely to see and to anticipate and/or respond to those attacks more effectively. The STIX Ontology utilizes twelve core concepts: Attack pattern, Campaign, Course of Action, Identity, Indicator, Intrusion Set, Malware, Observed Data, Report, Threat Actor, Tool and Vulnerability.

Attack Pattern describes ways that threat actors attempt to compromise targets, and *Campaign* categorizes malicious activities that occur over a period of time by identifying their intended targets. *Vulnerability* describes a flaw in software (or hardware) that can be exploited by a *Threat Actor* to breach a target.

Our objective in defining the CTI Ontology is to unify information from three sources (described earlier in this section) and facilitate logical reasoning about the security of CPS using *Axioms*. Axioms are rules used by a reasoner to infer additional information that may be hard to define in a knowledge representation language. To provide a perspective of the complexity of CTI Ontology, it includes 6657 axioms that describe CTI data. In addition to STIX, the our CTI Ontology also inherits characteristics from two additional Ontologies:

- **Cyber Observable Expression (CybOX)** - A standardized language for encoding and communicating information about cyber observables [20]. Using CybOX language [21], relevant observable events or properties pertaining to an attack pattern can be captured.
- **Common Attack Pattern and Enumeration (CAPEC)** - Provides a dictionary of known patterns of attack employed by adversaries to exploit known weaknesses in cyber-enabled capabilities [22].

E. MITRE ATT&CK Framework

The CTI Ontology obtains a perspective on the common techniques and tactics used by adversaries through the MITRE ATT&CK framework [23]. This information is useful while assessing an organization's cyber risk and to prioritize threat response. The framework, which stands for Adversarial Tactics, Techniques, and Common Knowledge, was officially released in May 2015 but has undergone several updates since then. Successful and comprehensive threat detection requires understanding common adversarial techniques and prioritize threats that may especially pose a severe risk to an organization, in addition to detecting and mitigating these attacks.

The ATT&CK framework is a comprehensive matrix of tactics and techniques. The aim of the framework is to improve post-compromise detection of adversaries by illustrating the actions an attacker may have taken. It is vital to understand how the attacker(s) gained access and how they migrate within a network. This framework helps identify those problem areas and contributes to the awareness of an organization's security posture at the perimeter and beyond. Organizations can use the framework to identify holes in defenses, and prioritize them based on risk.

ATT&CK can be extremely useful for evaluating an environment's level of visibility against targeted attacks with the existing tools deployed across an organization's endpoints. A technique is a specific behavior to achieve a goal and is often a single step in a string of activities employed to complete the attacker's overall mission. ATT&CK provides many details about each technique including a description, examples, references, and suggestions for mitigation and detection.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

A tactic is an objective or mission of an adversary. It describes what an attacker hopes to achieve with a specific compromise. Each tactic contains an array of techniques that have been used by malware or threat actor groups in known compromises. There are 11 tactics and over 250 techniques identified in the framework.

ATT&CK aids in the strategic response to cyber risks by outlining the tactics, techniques and attack vectors that could be used to compromise a CPS system. This insight, in addition to the structure of threat intelligence offered by STIX, may prove to be invaluable in identifying, enumerating, quantifying and addressing risks in CPS.

Here is a brief look at some of the important characteristics of our CTI Ontology:

- **Attack:** This feature is mapped to the *Indicator* and *Observed Data* classes in the STIX Ontology and the *Observation, FeatureOfInterest* and *ObservableProperty* classes in the STIX Ontology. This characterizes a cyber attack by identifying a pattern and a set of adversarial behaviors or information observed on a system in the network.
- **Exploit:** Mapped to the *Vulnerability* and *Intrusion set* classes in the STIX Ontology and the *Sensor, Actuator* and *Sample* classes in the SOSA Ontology, the Exploit feature enumerates a flaw in a platform (Software or Hardware with a CPE entry in the NVD) that can be leveraged by an adversary to compromise a CPS system.
- **Ramification:** Incident response teams often desire to know the consequences/objectives of potential adversaries to prioritize responses to cyber attacks. In a similar vein, threat modeling at the design phase of a CPS system will equip CPS designers to understand the outcome of cyber attacks and design more secure or resilient systems. At present, threat classification is based on the Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service and Elevation of Privilege (STRIDE) model [24], where each type of threat is assigned its own class. The *Ramification* feature maps to a class in the STRIDE based on the nature of the threat. In addition, it also maps to the *ThreatActor, CourseOfAction* and *Vulnerability* classes in the STIX Ontology and the *Actuation, Observation, Procedure, FeatureOfInterest, Platform* and *ObservableProperty* classes in the SOSA Ontology.

Thus, our framework allows users to identify and enumerate cyber threats that affect a CPS system of interest. We rely on Ontologies because of the following benefits they offer:

- **Knowledge Representation:** The primary benefit of using an Ontology is its ability to define a semantic model of data within the context of an associated knowledge domain. This can be leveraged to achieve knowledge sharing and, more importantly, knowledge reuse, which is discussed in the next section.
- **Logical Reasoning:** Reasoning in Ontologies and knowledge bases is an important property. Reasoning refers to deriving facts that are not explicitly specified in the Ontology. Ontologies use description logic to facilitate tractable reasoning.
- **Modularity:** Our framework facilitates modularity by allowing CPS designers to use domain-specific properties (Ontologies like SOSA). Users have the option of using additional vocabulary, in addition to the W3C specification to model proprietary systems.
- **Extensibility:** CPS systems are constantly evolving. Advances in networking and embedded system technologies like system-on-chip (SoC) and wireless transceivers result in the emergence of new CPS applications. The structure of SIMON, coupled with its modular design, supports integrating or modifying CPS characteristics, and facilitates reasoning about the security posture of a system.

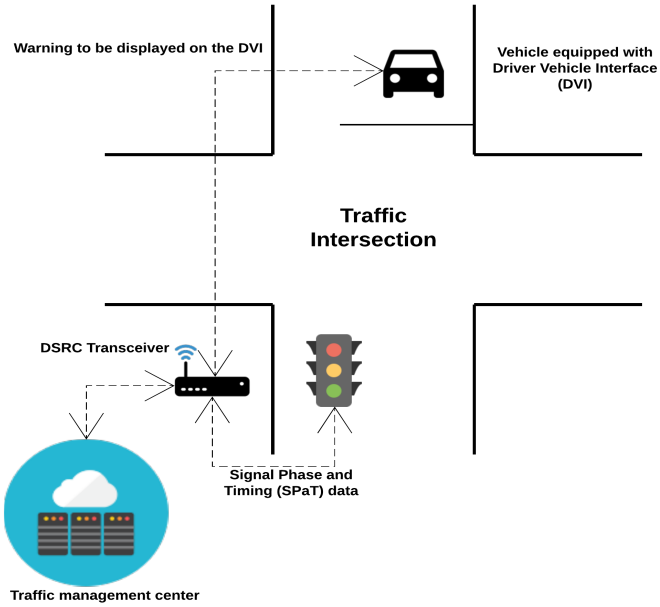


Figure 2. The RLVW system

IV. Vehicle To Infrastructure (V2I) Wireless Data Interface Ontology: A Case Study

As a case study to show the value of our framework, we use the Red Light Violation Warning (RLVW) safety application as described in the US Department of Transportation document [25]. The Red Light Violation Warning (RLVW) application enables a connected vehicle approaching an instrumented signalized intersection to receive information from the infrastructure regarding the signal timing and the geometry of the intersection. The application in the vehicle uses its speed and acceleration profile, along with the signal timing and geometry information, to determine if it appears likely that the vehicle will enter the intersection in violation of a traffic signal. If the violation seems likely to occur, a warning can be provided to the driver.

Figure 2 depicts the RLVW system. To identify the most vulnerable areas in this system, it is vital to understand the flow and origin of data (i.e., sensing and observation aspects of the system). Intelligent Transportation Systems (ITS) developers and automobile companies will be designing their CPS components to take advantage of the upcoming 5G data networks. Because such networks provide increased bandwidth and reduced latency, data will not only travel faster and in larger packets but will also be more vulnerable to attacks. For this case study, we developed an Ontology that highlights the data activity around the wireless portion of the RLVW protocol.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

In addition to choosing this region, we have highlighted only the CPS components that have either wireless capabilities or using the data collected from the wireless components. Thus, the Ontology highlights the wireless data interface portion of the V2I system where our conditions are met.

The design of the Ontology itself was made with respect to a few different factors. One of which was the component usage of the 5G networks. Components at the top of the hierarchy had active roles in communicating data from the Infrastructure to the Vehicle or vice versa. Components towards the bottom had more specific roles in acquiring and processing certain types of data that were necessary for signal phase calculations, optimal deceleration distance, and Differential Global Positioning System (DGPS) calculations. Organizing the Ontology with this factor in mind will allow for developers to quickly find the affected component during an attack based on the V2I data usage of the attacker. If the attacker had access to large amounts of data, it is highly likely that a component making heavy usage of the 5G network was involved. Conversely, if the attack was less threatening and had access to a smaller amount of data, the component involved would most likely be at the bottom of the hierarchy.

Another factor in designing the Ontology was splitting the components responsible for generating data into data collection and data calculation roles. Often times, attacks involve limiting the capabilities of components to collect data, whereas others involve altering the calculations of the collected data. To distinguish between the two, we organize components into cyber and physical categories. The cyber components are responsible for calculations, whereas the physical components (i.e sensors, actuators) are responsible for collecting data.

Lastly, we assign data types to each of the hardware and software components. Not only does this help understand which components are making use of which data, but it provides indicators in times of attacks. To elaborate, if it is known what type of data an attack is making use of, we can use a traceability methodology to start from the bottom of the Ontology at that specific data type, and trace up the Ontology until we find potential components that could be involved in the attack. Then, modifications and countermeasures can be taken to patch these vulnerabilities.

The flow of data in the Ontology has revealed that the Infrastructure Wireless Data System (IWDS) and the Vehicle Wireless Data System (VWDS), which are connected through the V2I Wireless Data Interface, are the most vulnerable regions of the entire V2I CPS, because in this data flows through an open network. With the source and destination IP addresses of data packets unprotected, this can lead to numerous threats from any third party with a V2X communication handler.

Now, we describe how our framework and the Ontologies described in Section III can be used to evaluate the RLVW system. Our framework, which extends NIST CPS framework and includes the Conceptualization phase, Abstract and Concrete Realization phases, and Assurance phase.

A. Conceptualization Phase

The design goal of the Vehicle to Infrastructure (V2I) Wireless Data Interface (WDI) system is to communicate relevant data between the Infrastructure and Vehicle application components through WDI and Application Platforms (APs). The V2I WDI incorporates algorithms and data exchanged to perform calculations to recognize “high-risk” situations. This inference results in issuing driver alerts and warnings through specific protocols. The most primitive and fundamental goal of the V2I WDI is to calculate and communicate Signal, Phase and Timing (SPaT) information to the vehicle with support of driving advisories and warnings [25]. The system is also responsible for maintaining authenticity of transmitted data through security measures. Corrupted data can result in compromising driver safety and information privacy. In our view, the three primary trustworthy design goals of the V2I WDI system are:

- **Verify Incoming Data (VID):** Since the system serves as a bridge between the vehicle and infrastructure domains, its main design goal revolves around transmitting data between both components. Therefore, a key requirement of this system is to verify the authenticity of incoming data from either side of the system, to avoid Phishing and other instances of fraudulent data transfer. This should be accomplished through ingress filtering protocols set in place to verify packet source headers and IP addresses.
- **Verify Outbound Data (VOD):** The WDI system is also responsible for generating advisories and alerts tailored to each nearby vehicle. With this in mind, a supporting requirement for this design goal must be to implement Secure Socket Layer (SSL) protocols or an alternative cryptographic key to ensure outbound data is not tampered with before reaching its destination.
- **Data Routing to Proximate Vehicles (DRPV):** Because this system is involved with establishing multiple connections between the infrastructure and vehicles, there is no generic set of messages purposed for all vehicles. Each advisory is calculated using metrics provided by each vehicle, thus creating a functional requirement to ensure that each message is sent to the appropriate vehicle. Failure of this requirement can serve fatal if metrics are sent to the incorrect vehicle, which may result in traffic violations or accidents.

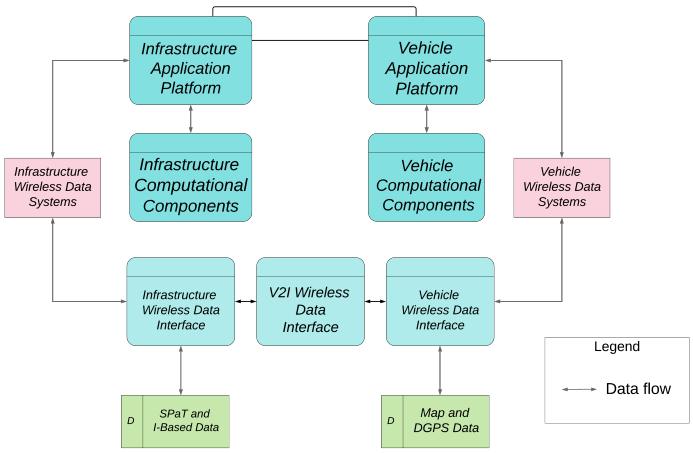


Figure 3. The V2I Wireless Data Systems Network

B. Abstract Realization Phase

The functional requirements listed in the conceptualization phase are purposed to describe the theoretical capabilities of

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

a CPS. When moving into the application layer components that satisfy the desired goals of the V2I WDI System, it is important to categorize each component along the respective requirement it resolves. This way, in the assurance phase, it can be tested how well the design goal of each component meets its functional requirement. Each component in the abstract realization phase will be assigned its own role.

Since the V2I WDI system is only a portion of the entire V2I domain, its design goal only covers data transmission. Therefore, only the transmission capabilities and roles of the categorized components will be discussed. Additionally, it is important to note that the sub components of both the infrastructure and vehicle contain similar components with only slightly varying goals. When working with CPS systems, the cyber and physical aspect of this CPS can be made resilient independently. However, the current issue that Intelligent Transportation System (ITS) developers face is maintaining that level of security when combining both sides of the system. This is because the integration of optimal designs when forming the system can lose the resiliency of both the cyber and physical aspects. To understand these challenges, we form a general hierarchy of the V2I WDI network that maps each component to the requirement it fulfils [25]. This will unravel the group of threats associated at each layer of the system. Figure 3 shows an overview of the V2I wireless data interconnect.

1) Verify Incoming Data (VID) Associated Components:

- **Infrastructure Wireless Data Systems (IWDS):** The Infrastructure Wireless Data Interface (IWDI) is responsible for sending and receiving data to/from nearby vehicles via the V2I Wireless Data Interface (VWDI). Its main role is to validate passing data by making sure position accuracy of incoming vehicles is up to the DoT standards. Additionally, the system calculates SPaT and Differential Global Positioning System (DGPS) metrics to be deployed to nearby vehicles via the IWDI.

  The IWDI role helps realize all activities related to communication with vehicles equipped with a VWDI. In other words, all three conceptual design goals are supported by the IWDI role. The conceptual design goals mandate that the security, privacy, and resiliency requirements be associated with the IWDI role.

- **Infrastructure Application Platform (IAP):** The IAP is the computational platform, which hosts the Infrastructure Application Component and provides the necessary hardware and software interfaces enabling communication with Infrastructure Wireless Data Systems, Infrastructure Data Systems, Roadside Signage System, Traffic Signal Controller, and Local/Back Office User Systems. Its main role is to channel all data gathered by sensors and physical systems to the cyber components. It can be considered the bridge between the cyber and physical components of the infrastructure side of the CPS, thus making it one of the least resilient and most vulnerable parts of the CPS.

  The IAP role is perhaps one of the most important in the RLVW system. It facilitates the interaction between the constituent systems in the infrastructure and the vehicle. It is apparent from the conceptual goals that the IAP role must meet the security, privacy, resiliency and reliability requirements.

- **Vehicle Wireless Data Systems (VWDS):** This component receives messages from the Vehicle Application Component through the Vehicle Application Platform, and formats and processes messages to be received by infrastructure components. This system also transmits data from the Vehicle Wireless Data Interface to the deeper hardware of the vehicle. This system also obtains GPS location and time. It may include a processor for GPS differential correction. Its main role is to convey information from the capture point at the Vehicle Wireless Data Interface to the internal components below and vice versa.

  The VWDS role is essential in ensuring communication between the sensors in the infrastructure space and the innards of VDWI. Hence, it must support the security and resiliency requirements outlined in the previous section.

- **Vehicle Application Platform (VAP):** The Vehicle Application Platform is the computational platform, which hosts the Vehicle Application Component and provides the necessary hardware and software interfaces enabling communication with Vehicle Wireless Data Systems, Vehicle Data Systems, and the Driver Warning Systems. Its main design goal is to channel all data gathered by vehicle sensors, actuators, and On-Board Diagnostics (OBD) data to the vehicular cyber components for processing and calculations. It can be considered as the counterpart to IAP on the infrastructure side. The security responsibilities of VAP are identical to those of IAP.

2) Verify Outbound Data (VOD) Associated Components:

- **Infrastructure Wireless Data Interface:** The IWDI is responsible for sending and receiving to nearby vehicles via the V2I Wireless Data Interface. Its main design goal is to refresh data transmission frequency at a configurable pace. It is also required to be equipped with countermeasures in case of corrupt or tampered data transmission. In these cases, it should issue warning messages to nearby vehicles to terminate data transmission and calculations using any information that comes from the Infrastructure.

  IWDI defines the functional requirements pertaining to communication with VWDI. The functional requirements of IWDI dictate that it should support security and resiliency.

- **Vehicle Wireless Data Interface:** The VWDI is responsible for sending and receiving to nearby Industrial Control Systems such via the V2I Wireless Data Interface. Its main design goal is to validate incoming data and request new packets from the infrastructure at a configurable frequency. It is also required to correct map and DGPS data for the infrastructure application component to produce the most precise RLVW metrics. In the case of inaccurate or corrupt data, the VWDI is required to terminate data transmission and issue alerts to the driver information interface.

  VWDI is the vehicle-side equivalent of IWDI. So, intuitively, this role should support the same security requirements as IWDI: security and privacy.

3) DRPV Associated Components:

- **V2I Wireless Data Interface:** Acts as a bridge for data transmission between the entire Infrastructure and

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Vehicle components. It receives raw data from the Infrastructure and vehicle components. This communication is functional over a bi-directional Dedicated Short Range Communication (DSRC) network. Therefore, its security protocol is effective within 1000 meters of any attacker. Beyond that, connectivity is loose and vulnerable. Its main design goal relative to the RLVW application is to ensure secure data transmission between approaching vehicles and signalized intersections.

It is evident from the description of this application that it sustains all three design goals of the RLVW system. Its vital importance means that this role should support privacy, reliability, resilience and security.

C. Infrastructure Data Types and Significance

Starting with the Infrastructure, its physical components consists of the signalized intersection sensor systems that capture two main types of data [25].

D. SPaT

SPaT data (Signal Phase and Timing) contains information about the behavior of the traffic controllers regarding the state of the signal (viz., red, green or yellow), how long that state will remain, and time until next phase change.

E. Driving Conditions

The physical component of the infrastructure also produces data that characterizes the environmental conditions approaching vehicles may face. This data consists of weather data, visibility data, and road conditions for the vehicle to incorporate in its decision making computations, to improve precision in judgement as approaching the intersection.

F. Vehicle Data Types and Significance

The vehicle's physical components consists of the position and stability systems, actuators, and telematic sensors that transmit Differential GPS (DGPS) and Dynamic Telematic Data (DTD) [25].

1) Differential GPS: DGPS data contains map data of the vehicle's position relative to the approaching signalized intersection. The vehicle data systems transmit DGPS to the infrastructure in order to alert the traffic controllers of the instantaneous distance the vehicle is from the intersection.

2) Dynamic Telematic Data: DTD consists of information regarding the vehicle's speed and position, and reveals how the vehicle is behaving internally. This data is combined with DGPS and incoming SPaT data from the vehicles to make calculations using DVI equations and algorithms in order to make a precise judgement of whether the driver should increase or decrease speed to avoid traffic violations and or accidents at the intersection.

G. Concrete Realization Phase

Now that the baseline for the design goals and supporting components are established, we can identify technical aspects of the identified components to understand how these functional requirements are met. Mapping the hardware and software to their respective components will help unravel the classification of security threats, since it is at this phase where the core data transmission occurs. Up until now, the above layers cover high-level understandings of the V2I WDI System. Now, we will classify core hardware and software that is generalized for both sides of the system in order to understand the mechanics behind V2I data transmission.

- **DSRC On Board Unit (OBU):** The DSRC OBU is the dedicated communication device installed on V2X connected vehicles. This hardware is responsible for establishing and receiving SPaT and Roadside data at a configurable frequency between 5.8 GHz -5.9 GHz. It utilizes the widely adaptive ThreadX RTOS operating system designed specifically for Internet of Things (IoT) applications. The DSRC OBU assists in enabling the capabilities of the Vehicle Wireless Data Interface [26].
  The OBU resides in vehicles and is responsible for implementing the VWDS, VAP and VWDI roles from the abstract realization phase. All of the security requirements associated with each constituent abstract-level component must be supported by the OBU. For example, an encrypted communication channel will fulfill both privacy and confidentiality requirements mandated by the roles that this component supports.
- **DSRC Roadside Unit (RSU):** The RSU unit performs identical functions but on the other end of the V2I wireless network. It is responsible for receiving SPaT and Roadside data from the infrastructure technical systems, verifying the data, and transmitting it upon data request from nearby vehicles. The RSU unit enables the capabilities of the V2I Wireless Data Interface, acting as the cyber bridge between the Vehicle and Infrastructure cyber components.
  The RSU is responsible for supporting the roles of IWDS, IAP, and IWDI. The security requirements associated with each of the three roles need to be supported by the RSU.
- **Wireless Sensor Network (WSN):** The WSN is the sensor network on the infrastructure side that captures road conditions data, infrastructure-based vehicle detection, road conditions, speed data, visibility data, and weather data. It utilizes sensors and actuators for the detection aspect of the hardware and standard transceivers, antennas, and receivers for the communication aspect of the hardware [27]. The Infrastructure Wireless Data Systems are supported by this WSN network, acting as the source of raw data that is formatted and processed into metrics by the Data Systems.

The WSN resides in the intersection between infrastructure and vehicle subsystems, and facilitates communication between the IWDI and VWDI systems. It is required to support the security requirements associated with these two roles.

H. Assurance Phase

The assurance phase deals with obtaining confidence that the CPS system built in the concrete realization phase satisfies the goals described in the abstract realization and conceptualization phases. Validating the concrete CPS system involves ensuring that it meets the functional and security requirements associated with the roles that each component supports.

Figure 4 illustrates the hierarchy of role allocation in SIMON. Evaluating the security posture of a CPS system requires current CTI data from multiple sources. To that end,

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

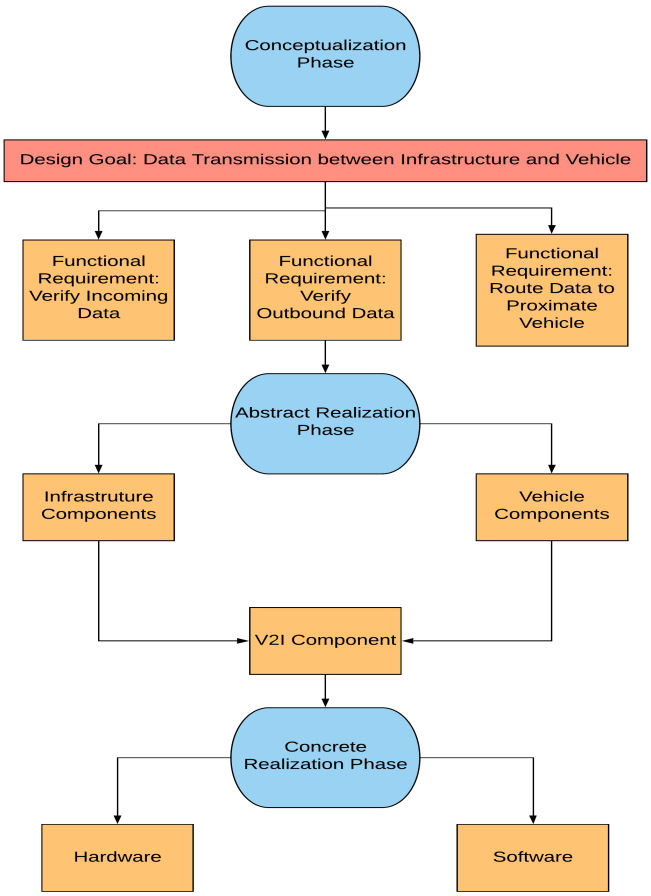


Figure 4. Role allocation hierarchy

SIMON's CTI Ontology discussed in Section III-D provides pertinent information.

Let us consider the example of an OBU running ThreadX RTOS. The OBU is responsible for sustaining the VWDS, VAP and VWDI roles, which necessitate the support for privacy, security and resiliency requirements. CTI is able to formulate a CPE identifier for this system using information obtained from the NVD. CPE:2.3:o:marvell:88w8997_firmware:-:*:*:*:*:*:*:* identifies the ThreadX-based firmware on a Marvell Avastar WiFi device. The Common Vulnerability Scoring System (CVSS) metrics from the NVD for this CPE indicate that the attack vector for a threat that exploits this vulnerability would be adjacent, which means that any infected devices in a local network could potentially compromise other devices in the network. Furthermore, the high severity score from the CVSS metrics indicates that a an attack that leverages this vulnerability could be catastrophic. If the system were to be affected by CVE-2019-6496 [28], an adversary may be able to launch a denial of service attack on the OBU. The vulnerability allows remote attackers to execute arbitrary code or cause a denial of service (block pool overflow) via malformed WiFi packets during identification of available WiFi networks. Exploitation of the WiFi device can lead to exploitation of the host application processor in some cases, but this depends on several factors, including host OS hardening and the availability of DMA.

To understand the impact of this vulnerability on the CPS system, the requirements traceability property offered by SIMON must be leveraged. This would show how the impact of a potential exploitation of this vulnerability would propagate up the three stages of design processes. Figure 5 shows various inferences that the reasoner makes in providing the insights presented below.

```
The Ontology is consistent
On-Board Unit (OBU) CPE : cpe:2.3:o:marvell:88w8997_firmware:-:*:*:*:*:*:*:*
(Asserted) OBU uses ThreadX OS
(Inferred) CVE-2019-6496 could be exploited
(Inferred) Potential violation of requirement 1.2.1 of the VWDS System
(Inferred) Potential violation of requirement 1.5.2.2 of the VAP system
(Inferred) Potential violation of requirement 1.4.2 of the VWDI system
```

Figure 5. DoS attack inference

- In the concrete realization phase, a vulnerability in the OBU would violate the functional requirements of both the DSRC roadside unit and the OBU. It is desirable to implement mitigative measures in the concrete realization phase because it wouldn't require a complete overhaul or re-engineering of systems previously implemented.
- In the abstract realization phase, all the roles fulfilled by the OBU and DSRC transceiver, VWDS, VAP, VWDI, IWDS, IAP, IWDI are violated. The corresponding security requirements pertaining to availability are affected. CVE-2019-6496, being a vulnerability exploited for DoS, confidentiality and integrity requirements may not be impacted.
- In the conceptualization phase, all three requirements (VOD, VID and DRPV) are affected by the unavailability of the OBU. thereby impacting the primary design goal of the RLVW system, which is to prevent roadway fatalities by ensuring date transmission between the infrastructure and vehicles.

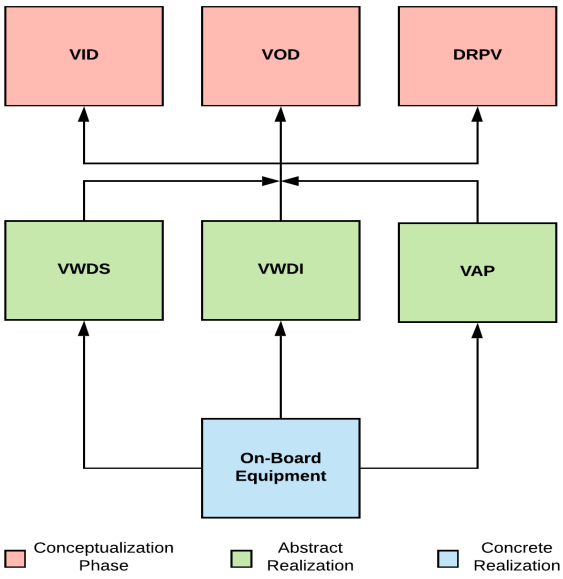


Figure 6. DoS attack on the OBU

A DoS attack on the OBU would violate the availability requirement for all three roles supported by the OBU (VWDS, VAP and VWDI), thereby violating the DRPV design principle of the CPS system. Figure 6 shows the how the design goals

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

of the RLVW system will be affected by such an attack on the OBU. The knowledge reuse property of SIMON can be used to compare various CPS systems to identify mitigative measures from other domains that can be reused in the CPS system under consideration. We have presented multiple examples in our prior work [1]. These insights would be invaluable to CPS system designers.

I. *Identifying Security Threats and Protection Mechanisms*

In this section, let us consider a few vulnerabilities and potential corrective measures in the RLVW system using SIMON. Now that the baseline for the V2I WDI region is set, we can analyze the proposed Ontology to classify potential threats in the flow of data.

1) V2X Remote DSRC Interjection Threat: The IWDS and VWDS communicate through the V2I WDI over a bidirectional DSRC network [25]. While DSRC provides a robust and low latency connection for short distance communication [29], its security protocol only prevents Distributed Denial of Service (DDoS) attacks from a short distance. Therefore, a third party with V2X communication handlers can interject data transmission remotely through Internet Protocol and Domain Name Service (IP/DNS) spoofing attacks to reroute outgoing Differential GPS (DGPS) data and Dynamic Telematic Data (DTD) from the vehicle. With this data in their possession, unauthorized V2X handlers can track drivers and read into vehicle logs, which creates privacy issues for the victim. The NIST Vulnerability Database highlights a similar issue with the configuration *cpe:2.3:a:cisco:application-policy-infrastructure- controller:8.31s6:*:*:*:*:*:*:** [17]. Existence of this vulnerability suggests that this simple attack is highly probable, if correct mitigation is not in place. A potential start for resolving this issue may involve ITS developers implementing a SSL certificate with outgoing data, which requires V2X handlers to have a certain cryptographic key in order to access the contents of the data packets [30].

```
The Ontology is consistent
Road side equipment CPE : cpe:2.3:a:cisco:application-policy-infrastructure-controller:8.31s6:*:*:*:*:*:*:*
Adversary may leverage CVE-2017-12352 to gian elevated privileges
Adversary may tamper with the RLVW warning data that is broadcast to vehicles
(Warning) Potential violation of functional requirement 1.1.5 of the Road side equipment
(Inferred) Potential violation of functional requirement 1.2.4.7 of the RLVW system
(Inferred) Potential violation of functional requirement 1.2.5.2 of the RLVW system
(Inferred) Potential violation of functional requirement 1.1 of the Driver Interface system
```

Figure 7. RLVW Inference.

The CTI Ontology obtains vulnerability information for components identified in the concrete realization phase using NIST CPE (Common Platform Enumeration) identifications. In this example, let us consider one vulnerability that can be exploited for privilege escalation with NIST Common Vulnerability Enumeration (CVE) identification, *CVE 2017-12352*, associated with the CISCO router with *cpe:2.3:a:cisco:application-policy-infrastructure-controller:8.31s6:*:*:*:*:*:*:** [17]. An adversary can exploit this vulnerability in certain system script files on Cisco Application Policy Infrastructure Controllers to gain elevated privileges and execute arbitrary commands with root privileges on an affected host operating system [31]. The vulnerability is due to insufficient validation of user-controlled input that is supplied to script files of an affected system [31]. A simple fix would be to install a software update for the application policy infrastructure controller. However, to demonstrate the capabilities of Ontological modeling and reasoning, we will assume that no software patches are available for this component.

Figure 7 shows how the CTI Ontology uses semantic reasoning to link vulnerabilities to the design goals identified during the conceptualization phase. While an elevation of privilege attack can lead to catastrophic failure of the affected system, we will focus on adversaries potentially spoofing their identities in this example.

The Extensible Authentication Protocol (EAP), a certificate-based authentication scheme, can validate the V2X handler that issues requests for DGPS and DTD data. This prevents most spoofing attacks.

```
The Ontology is consistent
Distinction identified between SSGA and RLVW VWDS
(Asserted) SSGA uses wireless message authentication scheme
(Inferred) EAP introduces latency
(Inferred) Potential violation of requirement 1.3.1 of the Driver Interface System
(Inferred) Potential violation of requirement 1.5.2.2 of the RLVE system
```

Figure 8. Comparing the Ontologies

Figure 8 illustrates how the message authentication scheme is capable of preventing the spoofing attack identified by the CTI Ontology. However, this scheme introduces latency, which may impact the timing requirement listed in the conceptualization phase of RLVW. Let us investigate if message authentication scheme is a viable solution for RLVW.

```
(Asserted) Timing Requirements 1.3.4.1 needs to be met
(Inferred) RLVW zone needs to be extended to 100 meters
(Asserted) DSRC radio has a maximum range of 120 meters
(Inferred) Additional requirements need to be added at abstract realization
```

Figure 9. Testing compliance

As evidenced from Figure 9, the Ontology determines that the RLVW requirement to warn drivers well in advance of a red light violation to provide ample stopping distance may be violated by the latency that is introduced by the authentication scheme. Furthermore, the Ontology also infers that the components used in this system are capable of supporting the timing requirement as the DSRC transceiver has a range of 120 meters. To address this, the Ontology recommends that the warning zone be increased from 80 meters before the intersection to 100 meters, which should provide ample time for EAP to authenticate the communication. A requirement needs to be added in the abstract realization phase to include an authentication scheme that also includes fail-safe measures if authentication is inconclusive. A domain expert needs to be consulted to ensure that all design goals are accurately captured in the SIMON framework.

2) V2X Handler Elevation of Priviledge Threat: Unfortunately, DSRC communication between V2I WDI and VWDS is not the only insecurity of the WDI region. The performance requirements set by the DoT do not mention any form of security over the functionality of the IWDS and VWDS [25]. In this section, we investigate the possibility of improving the resiliency of a CPS system against privilege escalation attacks by implementing a fail-safe mechanism. The proposed Ontology outlines the path of data through the Infrastructure Application Component (IAC) and Platform (IAP) that reveals no form of encryption on data produced by the physical components or verification when that data is transmitted through the cyber components. Therefore, V2X Handlers with identical communication functionality and IP addresses can

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

replace the role of the IWDS in the TCP handshake and give false acknowledgement to the IAP. V2X Handlers can then tamper with outbound SPaT and road data, which results in the vehicle application component producing false metrics. These metrics may result in a red light traffic violation or even roadside accidents. A similar vulnerability issue is noted with the configuration *cpe:2.3:o:cisco:ios-xe:16.10.1:*:*:*:*:*:*:** in the NIST Vulnerability Database [17], thus indicating the possibility of this threat occurring roadside. A general solution to this vulnerability can involve ITS developers implementing an ingress filtering protocol that requires the VWDS to check incoming data packets for their source headers, to ensure it matches the one of the origin and to reject the packet if it does not [30].

To authenticate entities within a network, Public Key Infrastructure (PKI) encryption may be used. This requires a Certifying Agency (CA) to generate and assign a public key to each component in the system. The CA is maintained by the DoT. The messages are authenticated using Message Authentication Code (MAC). PKI is a comprehensive security and authentication scheme requiring all entities to ensure confidentiality, integrity, non-repudiation and end-to-end monitoring and key life cycle management.

The CTI identifies the configuration of the V2X handler and maps it to *cpe:2.3:o:cisco:ios-xe:16.10.1:*:*:*:*:*:*:**. It is able to identify vulnerability *CVE 2019-1756* that can be leveraged by adversaries to launch an elevation of privilege attack to breach the communication channel between the IAC and IAP. A vulnerability in Cisco IOS XE Software could allow an authenticated, remote attacker to execute commands on the underlying Linux shell of an affected device with root privileges [32]. The vulnerability occurs because the affected software improperly sanitizes user-supplied input. An attacker who has valid administrator access to an affected device could exploit this vulnerability by supplying a username with a malicious payload in the web UI and subsequently making a request to a specific endpoint in the web UI. A successful exploit could allow the attacker to run arbitrary commands as the root user, allowing complete compromise of the system [32].

```
The Ontology is consistent
Road side equipment CPE : cpe:2.3:o:cisco:ios-xe:16.10.1:*:*:*:*:*:*:*
Adversary may leverage CVE-2019-1756 to gian elevated privileges by code injection
Adversary may tamper with SPaT data
(Asserted) Potential violation of functional requirement 1.1.5 of the Road side equipment
(Inferred) Potential violation of functional requirement 1.2.4.7 of the RLVW system
(Inferred) Potential violation of functional requirement 1.2.5.2 of the RLVW system
(Inferred) Potential violation of functional requirement 1.1 of the Driver Interface system
(Inferred) Potential violation of functional requirement 1.1.6 of the RLVW system
```

Figure 10. Elevation of Privilege Threat Inference

The potential impact of this vulnerability being exploited is shown in Figure 10. The framework is able to infer that the primary design goals of the RLVW application and the roadside equipment may be violated as a direct result of this vulnerability.

As discussed in the previous example, EAP and message authentication can be also be used in this example to protect the RLVW system. However, we are interested in identifying possible resiliency measures that can be employed by the RLVW system to protect against privilege escalation attack. To identify activities that can be used in the vehicle to detect spurious data from the infrastructure, let us consider an autonomous vehicle that is capable of perceiving the world around it.

We have defined a simple Ontology that models approximately 3118 attributes of an autonomous vehicle that includes driving actions like stop and go, a collision warning system, a lane change detection system, and so on. The insights provided by this Ontology can be used to prevent attacks like those discussed above by introducing resiliency into the design of the CPS system. The inference engine compares the RLVW system against three principles of a fully autonomous vehicle.

- **Sensing the world -** It is imperative for autonomous vehicles to possess the ability to perceive the world around them.
- **Conveying intent -** Assuming that other autonomous vehicles are present in the immediate vicinity, conveying intent such as lane change or impending change in driving action to other vehicles (and possibly pedestrians) is required.
- **Situational awareness -** Assigning a context to the information obtained by sensing the world is essential in making an informed decision. Comprehending events in the environment with respect to time and space is crucial.

```
1. Ensure vehicle meets requirement for sensing the world (1.2.1.1)
                - Cameras in the front windshield to detect traffic lights (Refer to requirement 1.3.3.2)
```

Figure 11. Measure to introduce resiliency into the RLVW system

The Ontology limits the inference to the design principle of sensing the world for the RLVW system as the other principles do not apply to it. Applying all three principles will negate the role of the infrastructure elements in this V2I system. To that end, the insights provided by the Ontology are shown in Figure 11.

While this is only a preliminary design of a specific region of the V2I CPS, the potential of an Ontology-based model is shown through the vulnerabilities it can classify. By describing various components through their roles, data types, and functionality, the Ontology can reason about new threats or vulnerabilities upon the addition of an unknown component to the system. If the properties of the unknown component, which in this case study is a V2X handler, become known, the Ontology can use reasoners to infer where this new component may interject by comparing properties of the new component with existing components in the CPS. When a match is found, the Ontology will classify the new component in a certain instance of the CPS. This knowledge can be used to implement new levels of security and mitigation in existing components to make it difficult for V2X handlers to either interject the CPS, or play the role of a component in the CPS [33].

V. Conclusion and Future Work

In this paper, we have presented an argument for modeling CPS using Ontologies. We also presented SIMON, a framework that is based on the NIST CPS framework but extends it in several ways. We have presented an extension to our previous work on CPS design validation using semantic inference. Reasoning about a CPS realization and validating that the realization does not violate functional as well trustworthiness goals is essential in improving the security posture of a CPS system. Currently, the SIMON framework is not capable of

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

automatically translating design goals into Ontological models. We are currently exploring the possibility of extending our work to support this function in the future.

We demonstrated that the role allocation Ontology is capable of delegating the functional and security requirements among subsystems at various design stages of a CPS system. It offers requirements traceability to understand the impact of a security threat in CPS. An RLVW system was used a case study to demonstrate the role allocation Ontology's capabilities. In the future, we intend to investigate other CPS domains. We use Ontologies during each design phase of the framework to check for compliance and provide recommendations by reusing knowledge. Increased traction in CPS adoption, their growing complexity, and heterogeneous nature necessitates accuracy in capturing the relationship between various components in a CPS. Reasoning about a CPS realization and validating that the realization does not violate functional as well as trustworthiness goals is essential in improving the security posture of a CPS system. The SIMON framework can aid in this process. We have only described the framework at a very high level, and we plan to integrate various Ontologies and reasoning engines in the near future. Although Ontologies are used extensively for knowledge representation in domains such as healthcare and bioinformatics, we aim to leverage their capabilities to define a domain-agnostic framework that can be extended to various CPS domains by attributing domain-specific properties (like SOSA).

ACKNOWLEDGEMENT

This research is supported in part by the NSF Net-centric Industry-University Cooperative Research Center at the University of North Texas and the industrial members of the center. The authors would also like to acknowledge the infrastructure and support provided by the Center for Agile Adaptive and Additive Manufacturing funded through State of Texas Appropriation (190405-105-805008-220).

REFERENCES

[1] R. Y. Venkata, R. Maheshwari, and K. Kavi, "SIMON: Semantic Inference Model for Security in CPS using Ontologies," *ICSEA-2019*, pp. 1–2, 2019.

[2] A. Humayed, J. Lin, F. Li, and B. Luo, "Cyber-physical systems security—a survey," *IEEE Internet of Things Journal*, vol. 4, no. 6, pp. 1802–1831, 2017.

[3] J. Prinsloo, S. Sinha, and B. von Solms, "A review of industry 4.0 manufacturing process security risks," *Applied Sciences*, vol. 9, p. 5105, Nov 2019.

[4] J. Giraldo, E. Sarkar, A. A. Cardenas, M. Maniatakos, and M. Kantarcioglu, "Security and privacy in cyber-physical systems: A survey of surveys," *IEEE Design Test*, vol. 34, no. 4, pp. 7–17, 2017.

[5] Y. Ashibani and Q. H. Mahmoud, "Cyber physical systems security: Analysis, challenges and solutions," *Computers Security*, vol. 68, pp. 81 – 97, 2017.

[6] "What are ontologies?." URL: {https://ontotext.com/knowledgehub/fundamentals/what-are-ontologies/[accessed: 2019-06-11]}.

[7] D. A. Wollman, M. A. Weiss, Y. Li-Baboud, E. R. Griffor, and M. J. Burns, "Framework for cyber-physical systems," *Special Publication (NIST SP) - 1500-203*, 2017.

[8] P. Kamongi, M. Gomathisankaran, and K. Kavi, "Nemesis: Automated architecture for threat modeling and risk assessment for cloud computing," 12 2014.

[9] P. Kamongi, S. Kotikela, K. Kavi, M. Gomathisankaran, and A. Singhal, "Vulcan: Vulnerability assessment framework for cloud computing," in *Proceedings of the 2013 IEEE 7th International Conference on Software Security and Reliability*, SERE '13, (Washington, DC, USA), pp. 218–226, IEEE Computer Society, 2013.

[10] K. Janowicz, A. Haller, S. J. D. Cox, D. L. Phuoc, and M. Lefrançois, "SOSA: A lightweight ontology for sensors, observations, samples, and actuators," *CoRR*, vol. abs/1805.09979, 2018.

[11] B. Mozzaquatro, C. Agostinho, D. Goncalves, J. Martins, and R. Jardim-Goncalves, "An ontology-based cybersecurity framework for the internet of things," *Sensors*, vol. 18, p. 3053, Sep 2018.

[12] S. Fenz, "An ontology- and bayesian-based approach for determining threat probabilities," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '11, (New York, NY, USA), pp. 344–354, ACM, 2011.

[13] D. Settas, A. Cerone, and S. Fenz, "Enhancing ontology-based antipattern detection using bayesian networks," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9041 – 9053, 2012.

[14] P. Gonzalez-Gil, J. A. Martinez, and A. F. Skarmeta, "Lightweight data-security ontology for iot," *Sensors*, vol. 20, p. 801, Feb 2020.

[15] P. Bhandari and Manpreet Singh Gujral, "Ontology based approach for perception of network security state," in *2014 Recent Advances in Engineering and Computational Sciences (RAECS)*, pp. 1–6, 2014.

[16] M. Iannacone, S. Bohn, G. Nakamura, J. Gerth, K. Huffer, R. Bridges, E. Ferragut, and J. Goodall, "Developing an ontology for cyber security knowledge graphs," pp. 1–4, 04 2015.

[17] "National Vulnerability Database." URL: https://nvd.nist.gov/ [accessed: 2019-06-11].

[18] "Exploit-DB." URL: https://www.exploit-db.com [accessed: 2019-06-20].

[19] "Metasploit-penetration testing framework." URL: https://www.metasploit.com/ [accessed: 2019-06-20].

[20] S. Barnum, "Standardizing cyber threat intelligence information with the structured threat information expression (stix)," *MITRE*, 2014.

[21] "Cyber Observable Expression." URL: https://cyboxproject.github.io [accessed: 2020-06-10].

[22] "Common Attack Pattern Enumeration and Classification (CAPEC)." URL: https://capec.mitre.org/ [accessed: 2019-07-02].

[23] "Mitre ATTCK Framework." URL: https://attack.mitre.org [accessed: 2020-07-21].

[24] Microsoft Corporation, "The STRIDE threat model." URL: https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20) [accessed: 2019-07-08].

[25] Department of Transportation, "Performance Requirements, Vol. 3, Red Light Violation Warning (RLVW)," *Vehicle-to-Infrastructure (V2I) Safety Applications*, pp. 1–68, 2015.

[26] "Vehicle to Infrastructure interaction (V2I)." URL: http://www.mogi.bme.hu/TAMOP/jarmurendszerek_iranyitasa_angol/math-ch09.html [accessed: 2019-09-19].

[27] D. Sánchez-Álvarez, M. Linaje, and F.-J. Rodríguez-Pérez, "A Framework to Design the Computational Load Distribution of Wireless Sensor Networks in Power Consumption Constrained Environments," *Sensors(Basel)*, pp. 2–5, 2018.

[28] "National Vulnerability Database." URL: https://nvd.nist.gov/vuln/detail/CVE-2019-6496 [accessed: 2019-06-11].

[29] "Dedicated short range communications (dsrc) service," 2019. URL: https://www.fcc.gov/wireless/bureau-divisions/mobility-division/dedicated-short-range-communications-dsrc-service [accessed: 2019-06-11].

[30] "Ddos glossary," 2019. URL: https://www.cloudflare.com/learning/ddos/glossary/ [accessed: 2019-06-11].

[31] Cisco, "Cisco application policy infrastructure controller local command injection and privilege escalation vulnerability," 2017. URL: = https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20171129-apic [accessed: 2019-07-22].

[32] Cisco, "Cisco ios xe software command injection vulnerability," *Cisco security advisory*, 2019. URL: = {https://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20190327-iosxe-cmdinject} [accessed: 2019-07-22].

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[33] R. Y. Venkata and K. Kavi, "An Ontology-Driven Framework for Security and Resiliency in Cyber Physical Systems," *The Thirteenth International Conference on Software Engineering Advances*, pp. 4–6, 2018.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

DFASC: Distributed Framework for Analytics Security in the Cloud

Mamadou H. Diallo, Christopher T. Graves, Michael August,
Kevin Groarke, Michael Holstrom, and Megan Kline

Naval Information Warfare Center Pacific, San Diego, CA USA
U.S. Department of Defense
Email: {mamadou.h.diallo, christopher.t.graves, michael.august,
kevin.groarke, michael.holstrom, megan.kline}@navy.mil

***Abstract*—Processing big data requires advanced technologies that can extract useful information from large scale data to support decision making. These advanced technologies are currently being offered in the form of analytic tools hosted in the cloud, and are being developed using different techniques such as artificial intelligence, machine learning, data mining, and statistical analysis. However, these tools are not very secure since the data they operate on must be in plaintext in the cloud, thereby leaving the data vulnerable to both insider and outsider attacks. To address these security issues when running data analytics in the cloud, we propose DFASC, a Distributed Framework for Analytics Security in the Cloud. At the core of the framework is homomorphic encryption (HE), which enables operations to be performed directly on encrypted data without using the private decryption key. Using HE, DFASC can distribute homomorphically encrypted data and analytics into the nodes of a distributed system and allow the analytics to operate on the encrypted data in each node. As a framework, DFASC provides mechanisms to enable the incorporation of HE libraries and data processing algorithms into the framework, which can than be used to implement analytic tools. A fundamental challenge with HE is its performance overhead due to the computationally intensive HE operations. This challenge of accelerating individual HE operations needs to be solved before secure big data processing in the cloud can be made practical. The distribution of the analytics not only improves the performance of the underlying analytic algorithms, it also helps to speed up the underlying HE operations. To enable the sharing of the encrypted data between parties in the cloud, DFASC incorporates a cryptographic key management infrastructure. To analyze feasibility of the framework, it was extended to implement a system that classifies images using a Neural Network algorithm. The experimental results show performance improvement of the system, including in HE operations, as the number of nodes in the cluster is increased.**

***Index Terms*—*Homomorphic Encryption*; *Cloud Computing*; *Privacy*; *Data Analytics*; *Data Sharing*.**

I. Introduction

Recent advances in communications and networking technology have revolutionized the way information systems are being developed and used. Cloud computing technology is a result of these advances and provides a computing paradigm with a large amount of computing power and storage resources. Advances in cryptography are also enabling the development of stronger security protocols for cloud-based systems [1]. The cloud computing market is growing at a rapid pace, and includes top cloud providers such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform. According to the Fortune Business Insight magazine, "The global cloud computing market size stood at USD 199.01 billion in 2019 and is projected to reach USD 760.98 billion by 2027" [2]. These resources are being used to develop information systems for individuals and organizations such as social media platforms, collaborative environments, and Internet of Things (IoT) based systems. These information systems are generating increasingly larger amounts of data as they are being used by individuals and organizations, resulting in a tremendous amount of data. For instance, the number of Internet users was estimated to be 2.4 billion in 2014, 3.4 billion in 2016, 3.7 billion in 2017, and 4.4 billion in June 2019. These high numbers of Internet users are reflected in the amount of data being generated on the Internet. For example, in 2019, Google reported 300,000 billion searches conducted worldwide daily and FaceBook over 4.3 billion messages posted daily.

Performing data analytics in the cloud is becoming increasingly significant for organizations of all types and sizes. These analytics are based on techniques such as artificial intelligence, data mining, machine learning or statistical methods [3], [4], [5]. Organizations are taking advantage of these analytic tools to gain powerful insights out of the ever-growing pools of organizational data. These cloud based data analytic tools are being developed for various application domains [6], [7], [8], [9]. However, there is also a growing concern about data security and privacy in cloud-based systems and applications that provide analytic tools [10], [11], [12], [13]. In particular, cybersecurity attackers are becoming more sophisticated, and attacks on data in large organizations are occurring more frequently [14].

The current cybersecurity vulnerabilities in the cloud stem from the fact that data is not protected while being manipulated by analytic tools in the cloud, which is inherited from the shortcomings of current cryptographic techniques for securing data. The recommended randomized encryption schemes, such as the Advanced Encryption Standard (AES) and Blowfish, provide strong protection of data in transit and at rest, but do not protect data in processing. This means that data needs to

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

be decrypted in memory before processing of the data can take place, which leaves the data vulnerable to attacks from both internal and external attackers.

There are many examples of both insider and outsider attacks on large organizations' information systems. The personal data of 77 million Sony users was leaked in 2011. Information from 38 million Adobe accounts was stolen in 2013. A major example of an insider attack was Edward Snowden's leakage of data from within the NSA in 2013. Data from 110 million Target customers was also hijacked in 2013. In 2015, US Office of Personal Management records for more than 21.5 million people were stolen by an outsider. Credit information on 143 million American, Canadian and British customers was stolen from Equifax in 2017. These attacks occurred because of a vulnerable attack surface within the information systems of these organizations.

Another challenge is how the data can be shared securely among parties in the cloud. For organizations outsourcing their data in the cloud, sharing the data is an important benefit. For instance, a medical system that manages patient health records can be deployed in the cloud and hospitals can collaboratively use the system to share the health records between them. This issue of data sharing in the cloud has been significantly studied and various techniques have been introduced. However, most of these techniques are based on PKI, which requires a Certificate Authority (CA) to manage the cryptographic keys. The CA is a third party entity and as such increases the complexity of building cloud systems. Deployment and maintenance of a PKI is complex and expensive, and certificate management can be challenging.

To address the shortcomings of existing standard cryptographic schemes, Homomorphic Encryption (HE) has been proposed [15]. HE schemes have revolutionized data security, as they enable computation to be performed directly on the encrypted data without needing the private decryption keys. Given ciphertexts as input, HE allows computation to be performed directly on the ciphertexts to generate encrypted results. When these encrypted results are then decrypted, they yield the correct plaintext answer for the computation as if it were performed entirely in plaintext. However, HE, while significantly improving data security in untrusted environments, comes with significant computation and storage overhead [16]. In general, the computational complexity of HE is orders of magnitude higher than that of standard operations on plaintext. A given ciphertext encoding is also much larger than its corresponding plaintext. Acceleration of HE is an active area of research, and great strides have been made to speed up the underlying HE operations. Note that more progress still needs to be made for HE to be practical for performing big data analysis. Even with its significant computational overhead, HE can still be practical today for certain types of application domains such as interactive applications [17].

In this paper, we introduce DFASC, a distributed framework for analytics security in the cloud. DFASC leverages HE to provide data security for cloud analytics not only in transit and at rest, but most importantly, when being processed. The framework is modularized and extensible to enable the incorporation of different types of HE schemes. The framework also provides mechanisms for incorporating data analytic tools that use HE schemes across the nodes of the distributed framework. This enables data analytic tools to operate directly on the encrypted data. To enable data sharing, the framework includes a cryptographic key management infrastructure based on the approach introduced in [18]. Using this approach, an organization can analyze data in the cloud and share the results with other organizations. Distributing analytic tool execution across the nodes of the framework speeds up the expensive operations of the HE schemes to improve the overall performance of the tools. The framework enables tool developers using various machine learning and data mining algorithms to use the framework to build analytic tools, in addition to enabling system developers to leverage these analytic tools within their applications. The secure systems developed based on the framework can then be made available to end-users to analyze their data securely and privately in the cloud. Having access to different analytics will enable end-users to trade off between the quality of the results of the data analysis and the time it takes to perform the analysis. Furthermore, end-users will have the ability to share their data with other parties securely and privately.

The paper is organized as follows. In Section II, we describe the challenges in processing data and our proposed solution. In Section III we describe our overall approach. In Section IV we present our approach for data sharing within the framework. In Section V we outline the data flow through the system. In Section VI we discuss our implementation of the framework and sample application. In Section VII we present the results of experiments performed on the system. In Section VIII we contrast our paper with related works. We end the paper with a conclusion and future work in Section IX.

II. Background

In this section, we take a look at how organizations make use of data analytics in the cloud and give an overview of HE, which can be used to provide data security in the cloud.

A. Data Analysis in the Cloud

Data analytics in the cloud, or cloud analytics, is a service model where data analytics are pushed to the cloud to take advantage of the cloud's resources. A hybrid model can also be used as a to allow analytics to be implemented partially on the client side and connected to cloud side to form an integral system, which can be scaled in the cloud as the need arises. In this way, a hybrid approach can be used to split the data analysis between the client and the server. These analytics are developed using techniques such as artificial intelligence, machine learning, data mining, and statistical modeling and analysis.

Due to the benefits they are providing to organizations of all types and sizes, cloud analytics are gaining popularity. They

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

are steadily making their way into enterprise applications in the cloud in various areas, such as customer support, fraud detection, and business intelligence [6]. As organizations are becoming aware of these cloud analytics, the need for them is increasing. The major cloud service providers are responding to this need for tools that provide data analysis and business intelligence capabilities within the cloud by adding these features to their cloud services [19]. Thus, the trend of organizations outsourcing their Information Technology operations to the cloud, combined with the trend of cloud service providers adding more intelligence to their cloud services, indicates that increasingly organizations will make use of the cloud to analyze their large and potentially sensitive data sets.

However, the cloud is vulnerable to cyberattacks from both internal and external attackers, and running analytics in the cloud on organizationally sensitive data can result in loss of data security. In [20], an analysis of the security threats in big data analysis using MapReduce and Hadoop revealed the complexity of securing cloud analytics. One security challenge is related to the large amount of data to be processed, which makes current security techniques impractical because they are too slow to be effective [21]. Another security challenge is related to the various sources of data to be combined and processed in the cloud [22]. To address these vulnerabilities of the cloud, we use HE to ensure the confidentiality of the data that are collected, stored, and processed in the cloud.

B. Homomorphic Encryption Schemes

Homomorphic Encryption (HE) is a cryptographic scheme that enables operations to be performed directly on encrypted data without using decryption keys. The HE computations are represented as either Boolean or arithmetic circuits, which are characterized by their depth. There are different types of HE schemes based on the types of operations they support. *Partially Homomorphic Encryption (PHE)* schemes support only one type of operation, addition or multiplication, while *Fully Homomorphic Encryption (FHE)* schemes allow arbitrary computation on ciphertexts. Between these two extreme cases, there are schemes that are *Somewhat Homomorphic Encryption (SHE)*, which support both addition and multiplication but are limited in the number of operations (the depth of the circuit) that can be executed on ciphertexts.

The existence of fully homomorphic encryption was theorized in 1978 by Rivest [23]. However, it was only in 2009 that the first working *Fully Homomorphic Encryption* (FHE) scheme was constructed by Gentry [15]. Gentry's approach involves taking a *Somewhat Homomorphic Encryption* scheme and "squashing" the decryption circuit to reduce the noise in a process called "bootstrapping" to get the *Fully Homomorphic Encryption*. The security of his scheme assumes the hardness of two problems: certain worst-case problems over ideal lattices, and the sparse subset sum problem. However, this process was impractical due to the required computation time. Since then, a number of more practical FHE schemes have been proposed.

a) FHE Scheme Based on Ring Learning With Errors: Most of the FHE schemes that have been proposed base their security on the hardness of the (Ring) Learning With Errors (RLWE) problem [24]. The RLWE problem has been proven to provide a strong security guarantee while supporting more practical FHE schemes. Before describing some of these FHE schemes, let us first define the RLWE problem.

Definition of RLWE: let $n = 2^k$ and choose a prime modulus q such that $q \equiv 1 \mod 2n$. Let the ring $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$, represent the set of all the polynomials over the finite field $\mathbb{Z}_q$ for which $x^n \equiv -1$. Given samples of the form $(\mathbf{a}, \mathbf{b} = \mathbf{a} \times \mathbf{s} + \mathbf{e}) \in R_q \times R_q$ where $\mathbf{s} \in R_q$ is a fixed secret vector, an element $\mathbf{a} \in R_q$ is chosen uniformly, and $\mathbf{e}$ is chosen randomly from an error distribution in R_q. Given this definition of the RLWE problem, finding s is infeasible.

Using the RLWE problem, a message $m \in R_q$ can be encrypted by using the b element above as a one-time pad encryption scheme [25]. The ciphertext can be represented by $\mathbf{c} = \mathbf{b} + \mathbf{m}$, where $\mathbf{c} \in R_q$.

BGV Scheme Brakerski, Gentry, and Vaikuntanathan proposed a Leveled FHE scheme based on the RLWE problem and referred to the scheme as BGV [26]. It is referred to as "leveled" due to the fact that its parameters depend (polynomially) on the depth of the circuits that it is capable of evaluating. "Leveled" FHE means that the size of the public key is linear in the depth of the circuits that the scheme can evaluate, that is, its size is not constant. The key operation in the scheme is the *REFRESH* procedure, which switches the moduli of the lattice structure and switches the key.

C. Homomorphic Encryption Libraries

Following the constructions of the FHE schemes, software libraries implementing the schemes are being developed. The first of such libraries to be implemented is HElib, first released in 2012 [27]. This first version of HElib implemented only the Brakerski-Gentry-Vaikuntanathan (BGV) scheme, but the latest version now includes the approximate FHE scheme proposed by Cheon, Kim, Kim and Song (CKKS) [28], the newest FHE scheme. Following HElib, a number of other FHE development efforts have been launched as presented in the survey of the current FHE libraries in [29]. Here we limit our discussion to PALISADE [30] and Microsoft's Simple Encrypted Arithmetic Library (SEAL) [31].

The PALISADE library is being developed under an open-source project that provides efficient implementations of lattice cryptography building blocks and leading homomorphic encryption schemes. PALISADE is designed for usability, providing simpler APIs, modularity and cross-platform support. The current version of PALISADE supports the BGV, Brakerski-Fan-Vercauteren (BFV) [32], CKKS, and FHEW schemes and a more secure variant of the TFHE scheme, including bootstrapping [30]. The Microsoft SEAL is also an open-source library that provides an efficient implementation of lattice cryptography using leading homomorphic encryption

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

schemes. The current version of SEAL also supports the BGV, BFV, and CKKS schemes. The proposed DFASC framework integrates these two libraries: PALISADE and SEAL.

III. Framework

In this section, we describe the architecture of the proposed DFASC framework, the integration of HE libraries and data processing algorithms into the framework, the data sharing protocol used to enable clients to share encrypted data, and the threat model of the framework.

A. Architecture

The DFASC framework is designed using a hybrid client-server/distributed model, where clients send requests to a remote server, and the server forwards the client's requests to a distributed system for processing. The results of the data processing are returned to the remote server for storage and to be made available to the clients. The high-level design of the framework is presented in Figure 1. The architecture is composed of two main components: *Trusted Client* and *Untrusted Cloud Environment*.

The *Trusted Client* comprises three main sub components, *Client Manager*, *HE Manager*, and *Configurations Manager*. The *Client Manager* coordinates the activities of the client and manages the interactions with the server. The *HE Manager* provides support for HE operations including generation and storage of public and private keys, encryption and decryption of data, and keys revocation. The *Configurations Manager* keeps track of the cloud resources for the clients, which change dynamically as the system is being used. Note that system developers will need to extend the framework to build concrete systems for specific application domains. In addition to the above core components, system developers need to implement a user interface for end-users to interact with the system.

The *Untrusted Cloud Environment* is composed of an *Untrusted Server* and an *Untrusted Distributed System*. All the data sets sent by the clients to the *Untrusted Cloud Environment* will remain encrypted at all times. The sub-components of the *Untrusted Server* include a *Service Engine* for coordinating all the activities related to distributing data and operations into the *Untrusted Distributed System*; an *HE Manager* for managing HE libraries stored in the *HE Libraries* storage; an *Analytics Manager* for managing the analytic algorithms persisted in the *Libraries* storage; a *Sharing Manager* for sharing encrypted data between the clients; and a *Configurations* storage for storing various cloud configurations and metadata. The *Service Engine* communicates with the *Untrusted Distributed System* to coordinate its activities, including sending workloads and partitioning the nodes within the cluster.

The *Untrusted Distributed System* provides the infrastructure for distributing analytics algorithms. The inputs to the *Untrusted Distributed System* include the set of data to be processed and the software program to be executed on the nodes of the distributed system that will process the data. At the core of the distributed system is a *Distribution Manager*, which provides the mechanisms for generating the clusters of distributed nodes. The nodes are generated by the *Distribution Manager* on demand based on the configurations provided by developers. In addition, the *Untrusted Distributed System* provides an interface to enable interaction with other distributed systems.

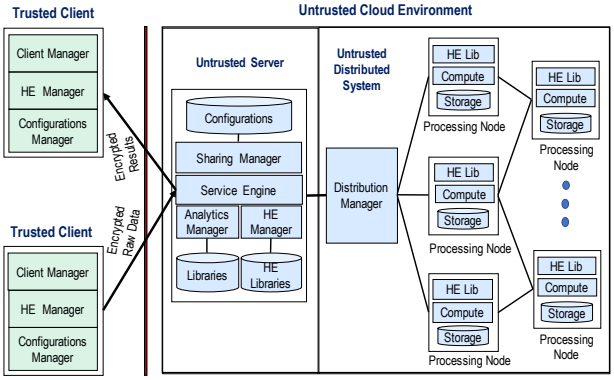


Fig. 1: Distributed Framework Architecture

B. HE Library Integration

At the core of DFASC is the mechanism for incorporating HE libraries into the framework. Like the standard cryptographic algorithms, homomorphic encryption algorithms have a well-defined set of operations. These operations include key generation, encryption, decryption, and ciphertext operations. To accommodate various FHE libraries, DFASC abstracts out the common core operations of HE libraries and builds those operations into the framework. It adopts a parameterization approach to enable each library to provide all necessary parameters to execute the operations. At a low-level of the implementation, a binary operation takes as inputs two integers A and B, and returns the result as an integer C. These operations are abstracted out into an interface that can then be used to integrate a given HE library. As part of the framework, we integrated the PALISADE and SEAL libraries.

C. Data Processing Algorithm Integration

DFASC provides an extensible interface to enable developers to extend or customize DFASC to add new machine learning and data mining algorithms into the framework. Considering the complexity of using existing HE libraries, the first machine learning algorithm we considered for the DFASC framework is the linear Support Vector Machine (SVM). The second machine learning algorithm we implemented within the framework was an artificial neural network. In the future, we plan on adding more machine learning algorithms into the framework.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

1) Support Vector Machines: SVMs are supervised learning models that can be used to analyze data based on classification and regression analysis. The SVM serves as a non-probabilistic binary linear classifier.

Consider a set S of sample data elements, and two subsets S_A and S_B of S, where $S_A \cup S_B = S$, and each element of S ($S_1 \in S$) is annotated as belonging to S_A or S_B. The SVM training algorithm generates a mathematical model that can be used to categorize new elements of S as belonging to S_A or S_B.

First, we are given a labeled training dataset of n points of the form $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$. This training dataset contains both the inputs and the desired outputs. Given the training dataset, we then compute the SVM model to be used for classification. This model then separates the elements of S into two classes, S_A and S_B, based on the classifier that was generated from the training data. The internal operations of the linear SVM include the dot product of vectors, addition, and subtraction. To demonstrate the utility of the DFASC framework, we implemented an SVM classifier on top of our distributed framework using the PALISADE library.

2) Neural Networks: Neural networks (NN) are a learning mechanism that model the biological brain. They consist of a set of transformations of a signal vector throughout a graph of nodes. Each node, called a neuron, is connected to the next layer of neurons via edges, called links. Each link has a weight associated with it. Each neuron processes its input signal as a linear combination of the weights of its input neurons according to an activation function and produces an output signal that gets forwarded to neurons in the next layer of the network. The training phase constructs a model by updating the weights associated with each neuron in the network. After being constructed, the model can be represented by a mathematical function and used to classify real world inputs to the neural network. In this way, neural networks are considered a black box machine learning approach. Deep neural networks typically have many layers and utilize specialized neural network architectures. Over the past few years there has been a resurgence in neural network research due to the success of deep neural networks when applied to certain application domains such as object detection and image classification. To demonstrate the applicability of the DFASC framework to parallelize an encrypted image classification task, we implemented a feedforward neural network classifier on top of our distributed framework using a homomorphic encryption library.

D. Security Model

In this section, we describe the threat model and security properties of the framework.

1) Threat Model: We adopt the *honest-but-curious* adversarial model. We assume that the client-side is trusted while the cloud environment is untrusted. We assume that Cloud Service Providers (CSP) as well as users can act as adversaries. When users send data into the cloud, the CSP has the ability to store the data in different locations and make use of it without the user's knowledge. We assume that the CSP will not deliberately tamper with users' data by inserting, deleting, modifying, and truncating parts of the data. An adversarial CSP may not provide false answers in response to user queries. We assume that the adversarial CSP cannot obtain a user's secret keys.

2) Security Properties: In the DFASC framework all users are required to register in the system to get credentials for authentication. Only users verified through authentication can gain access to the system. In addition to authentication, the framework employs a policy-based authorization service to provide access control to data. Using this service, users can decide who can get access to what parts of their encrypted data in the cloud. Homomorphic encryption schemes have been proven to be very secure. All the private keys for decrypting the data remain with clients, and only public keys are sent to the cloud.

IV. Data Sharing

One challenge in sharing data securely in the cloud is how to enable recipients to access the shared data. Many different techniques and approaches have been proposed in the literature to address this challenge [33], [34], [35]. Most of these approaches are based on the Public Key Infrastructure (PKI) technology, which is used to authenticate users and devices against information systems. PKI relies on a CA, which acts as a trusted third party responsible for managing and certifying public keys ownership. The CA associates a given user ID with a public key by generating a signature referred to as a certificate. In this approach, all users of a system can exchange their public keys for the purpose of data sharing. In this case, a $Sender$ wanting to share data with a $Recipient$ would use the public key of the $Recipient$ to encrypt the data. The $Recipient$ would use the corresponding secret key to decrypt the data. Through the use of digital signatures, the CA can guarantee that public keys will not be subject to impersonation, where a malicious party could replace the public key of a legitimate party with a compromised one.

One drawback of the PKI based data sharing is that it requires complex computations for data encryption and decryption, which can slow down systems with extensive data sharing. Another drawback is the dependency on the trusted third party CA, which increases the communication overhead in a system. To address these challenges, various techniques that combine public key and symmetric key cryptography have been proposed. For these approaches, the symmetric keys are used to encrypt and decrypt data and the PKI system is used to share the symmetric keys between the users. All sharing approaches based on PKI are exposed to the security vulnerabilities associated with a CA. If a CA is breached, the certificates can be compromised resulting in sending the data to the wrong users.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

In this paper, we adopt a simple approach proposed in [18] for data sharing in untrusted environments, which does not require a CA. The approach makes use of a key management system based on PKI to provide clients with mechanisms to generate, store, distribute, and revoke public/private keys in a distributed system. Instead of using a CA, our approach is to exchange the private keys using an email infrastructure, where each client is equipped with a built-in email server.

1) Access Control to Encrypted Data: There are various identity and access management (IdAM) systems that can be used to restrict access to resources in a system. Among other functionalities, these systems manage, identify, authenticate, and authorize individuals to ensure appropriate access to resources.

2) Data Partitioning: To facilitate data sharing, each client needs to partition its data based on its sharing policies. Each partition will be encrypted using a different public/secret key pair to restrict access to the data. The sharing policies define how the data can be partitioned in such a way that the number of keys required to encrypt the data is minimized. Let us denote $\{c_1, c_2, \ldots, c_m\}$, the set of all clients in the system. Let us also denote $\{d_1^{c_i}, d_2^{c_i}, \ldots, d_n^{c_i}\}$, the set of data partitions for a given client c_i. Then, for each data partition $d_j^{c_i}$, a public/secret key pair, $(pk_j^{c_i}, sk_j^{c_i})$, will be generated to encrypt $d_j^{c_i}$. This will give the client a flexible approach for sharing their data in the cloud at a fine-grained level of access control.

3) Exchanging Public Keys: For the purpose of sharing encryption keys, each client will create a sharing public/secret key pair (sk_{c_i}, pk_{c_i}). The first time two clients, c_i and c_j, interact in the distributed system, they exchange their public keys as follows. The client c_i sends a message to c_j containing the tuple (Id_{c_i}, pk_{c_i}), where Id_{c_i} represents the unique indentier of c_i, and the client c_j replies with a message containing the tuple (Id_{c_j}, pk_{c_j}).

4) Sharing Data: When a sender c_i wants to share a data partition $d_j^{c_i}$ with a receiver c_j in the distributed system, the sender needs to provide the receiver with the secret key $sk_j^{c_i}$ corresponding to $pk_j^{c_i}$ used to encrypt the data partition $d_j^{c_i}$ in order to decrypt it. To protect the secret key, the sender encrypts it using the receiver's sharing public key. The sender replies with the following message containing the tuple $(Id_{c_i}, Enc(sk_j^{c_i}, pk_{c_j}))$, where $Enc(sk_j^{c_i}, pk_{c_j})$ means that the $sk_j^{c_i}$ is encrypted using the pk_{c_j}. This will guarantee that only the intended receiver can decrypt the message containing the secret key.

Figure 2 shows an example where two clients, c_1 and c_2, have generated public/secret key pairs, (pk_{c_1}, sk_{c_1}) and (pk_{c_2}, sk_{c_2}), for sharing secret encryption keys, partitioned their data, and generated different public/secret key pairs to encrypt each partition separately. The sharing keys for both clients are published in the cloud through the *Sharing Manager* and the homomorphically encrypted data stored in the cloud through the *Storage Manager*. The example also shows the partition $d_1^{c_1}$ and its corresponding secret key $sk_1^{c_1}$, to be used to decrypt it, received by the client c_2 and shared by the client c_1.

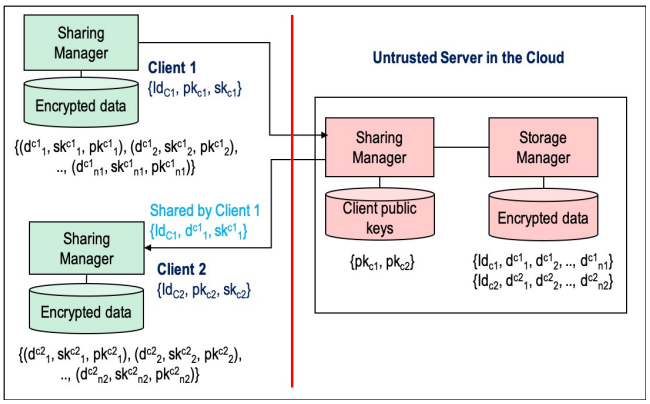


Fig. 2: Data Sharing Protocol

V. DFASC Operational Flows

The architecture of the DFASC framework comprises a number of components that interact to support the functionalities of the framework from the perspective of both developers and end-users. It abstracts out the complexity related to building a web-based client-server application, building a cloud-based distributed system, and connecting the two entities. In the following sections, we describe the operational flows of the framework, focusing particularly on how developers can extend the core components of the framework and instantiate it to build concrete systems, and then discuss how end-users can use those concrete systems.

A. Extending the Framework

For developers extending the framework, there are two main features: adding a new HE library, and adding a new data processing algorithm based on machine learning or data mining techniques. At the design level, the framework employs a modular design to isolate the HE libraries and data processing algorithms. At the implementation level, the framework uses containers to enable each HE library and each data processing algorithm to be self-contained. To add an HE library, the developer needs to deploy the HE library in a container and expose an API to enable the HE manager to make use of it. Similarly, a new data processing algorithm needs to be implemented and made available to the analytics manager, which will distribute it to the nodes at runtime. Both SVM and Neural Network implementations are included in the framework to serve as a guideline for developers to incorporate their own algorithms into the framework.

B. Instantiating the Framework to Build A Concrete System

The framework provides building blocks that can be used to build concrete distributed systems where analytic tools can be run in the encrypted domain. The application domain will determine the specific analytic tools to be applied using one

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

of the available HE-enabled machine learning or data mining algorithms. For instance, in the application we built to evaluate the framework, both an SVM and a Neural Network were determined to be suitable for an image classification task. During the analysis, each data point falls in one of ten classes. The application domain dictates the type of data and how it needs to be encoded appropriately to ensure compatibility with the data format of the underlying HE library. Recall that the current HE libraries support only low level operations, such as addition or multiplication of numbers. It is the task of the developer to figure out how the specific data types of the application domain can be transformed in such a way that these low level operations of HE can be applied to the data.

C. Using the Concrete System

Once the system is completed, then it can be made available to end users. There are two main workflows of the system for the end user: 1) analyzing data using an analytic tool, and 2) sharing data with other users. At a high-level, the following operational workflow depicts the process for analyzing data in the distributed system.

- The User opens the Client web-based GUI.
- From the Client GUI, the user uploads the raw data to the Client local storage.
- The User selects the analytic tool to be used to process the raw data.
- The User requests the data to be encrypted.
- The Client Engine selects the appropriate HE library, and uses it to encrypt the data.
- The Client Engine sends the encrypted data along with the user parameters to the Untrusted Server.
- The Untrusted Server selects the number of nodes to use in the distributed system.
- The Untrusted Server partitions the data according to the parameters selected by the user and pushes it to the nodes.
- The Untrusted Server notifies the User after the data has been distributed.
- The User requests data to be processed and forwarded to the Untrusted Server.
- The Untrusted Server delegates the workload to the Distribution Manager.
- The Distribution Manager initiates the data processing throughout the Untrusted Distributed System.
- After the execution is completed, the Untrusted Server gathers the results from the Distribution Manager, and sends them to the User.
- The Client Engine decrypts the results and displays them on the GUI.

The following operational workflow summarizes the process for sharing data in the distributed system. The Sharing Manager on the Untrusted Server is responsible for sharing encrypted data and encrypted secret keys between parties sharing data with each other. If the recipient does not already have the secret key to decrypt the data, then the Sharing Manager will request the secret key from the sender, and the sender will encrypt the secret key using the recipient's sharing public key and send it to the Sharing Manager, which serves as the proxy between sender and receiver. We assume that the user possesses a public/secret key pair to be used by the underlying sharing protocol. We assume that each party has the sharing public key of the receiver. We also assume the data to be shared is stored with the Storage Manager component.

- From the Client GUI, the sender selects the set of data to be shared, the recipients and their sharing public keys.
- The User sends the request to share the data to the Untrusted Server.
- The Sharing Manager on the Untrusted Server passes a message to the recipient containing a reference to the stored encrypted data.
- The Sharing Manager notifies the recipients about the availability of the data.
- The Recipients retrieve the shared data and use their secret keys to decrypt the data.

VI. Implementation

In this section, we describe the implementation of the DFASC framework including all the core components. We also describe the implementation of a use case system that instantiates the framework, referred to as Image Classifier. The Image Classifier system includes two data analytics for classifying images. The first analytic tool is implemented using SVM while the second analytic is implemented using NN. As described previously, both SVM and NN are integrated into the framework. This Image Classifier is used to evaluate the feasibility of the framework. We leveraged a number of open-source projects for the implementation including the Django web framework [36], Apache Hadoop [37], Apache Spark [38], and Xen hypervisor [39].

A. Framework Implementation

The implementation of the framework is broken down into three main subsystems: Client, Cloud, and Distributed Computation.

The Client subsystem is implemented as a web service, which includes a template for the web-based client interface, a web server for managing all the client services, and a database for storing encryption/decryption keys, plaintext data, ciphertext, and cloud configuration. We used the Django web framework to implement this Client subsystem to connect all the client modules. The Django Rest Framework allows for quick development of web based REST APIs.

The Cloud subsystem implements various modules corresponding to the core component of the framework to support its various services in the cloud. These services include managing HE libraries, data processing algorithms, analytics, cloud configurations, and data sharing among users. REST APIs allow developers to extend DFASC to build concrete applications, such as adding new HE libraries and data processing algorithms.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

We used Apache Spark as the basis to implement the distributed system. Spark is highly modularized, which simplifies its integration with other systems. Spark is an ideal distribution framework for DFASC, as it enables the distribution of data as well as programs for execution on the cloud nodes.

As mentioned previously, we selected the PALISADE and SEAL HE libraries as the first libraries to be integrated with the DFASC framework. Both PALISADE and SEAL are implemented using C++ and provide a simple interface to access their basic functionality. The integration of these HE libraries into our framework required building a C++ wrapper to interact with the Django web server written in Python as well as the Spark interfaces used for the distribution.

We used the Xen hypervisor to deploy a local instance of a cloud infrastructure as a service (IaaS). This local cloud serves as the testbed to generate and manage virtual machines for the distributed system. We used this local cloud instance to deploy and test our distributed framework.

B. Use Case: Image Classification

To analyze the feasibility and performance of the overall DFASC framework, we designed and implemented an image classification system using the framework. Image classification deals with labeling of images into predefined classes and training a classifier to classify a given image in one of those classes. Various machine learning classifiers such as SVM, K-Nearest Neighbors, and Decision Tree, or deep learning classifiers like Convolutional Neural Networks and Artificial Neural Networks, can be used to classify images. Image classification is useful in various application domains including autonomous driving, labeling x-ray images, and recognizing human faces for security purposes.

For our image classification system, we used the two available machine learning algorithms in the DFASC framework, SVM and NN, to implement two classifiers. As mentioned previously, DFASC includes two versions for each of the machine learning algorithms, corresponding to PALISADE and SEAL. In the following section, we describe the implementation of NN. To learn how we implemented SVM, refer to our prior publication [1].

1) NN Implementation: The specific NN we implemented is the Feedforward Neural Network, which we trained on the *sklearn Digits Dataset* [40], a reduced version of the MNIST handwritten digit dataset. Each input to the network is a 64 value array representing one of the 8 by 8 images in the dataset. The network was trained on plaintext data and then adapted to predict on encrypted data. The neural network consists of 4 layers. The input layer contains 64 neurons, with each neuron taking in one value from the 64-value input array. The second and third layers each consist of 128 neurons. This value was selected because it provided accurate prediction results on plaintext data but can be changed to optimize the calculation speed and accuracy of encrypted prediction results. The fourth and final layer is our output layer. The neuron with the highest activation in this layer is the Neural Network's final prediction. The sigmoid activation function was chosen since it can be adapted to operate on encrypted functions by representing it as a polynomial function. The following is the approximation function used:

$$f(x) = 0.500781 + 0.14670403x + 0.001198x^2 - 0.001006x^3$$

This function is described in [41]. We use cross-entropy to measure the distance between the predicted and actual probability distributions, which is then used in back- propagation to adjust the network's weights and biases. Once the network is trained, encrypted data can be fed through the network using the sigmoid approximation function described above. The output can then be decrypted to see the network's prediction. In order to speed up the calculations, multiple inputs are distributed across the Spark cluster such that each node performs one prediction and returns the results.

2) Implementation of Image Classification Application: Reusing the Software Defined Radios Link Analyzer system we introduced in [1], we implemented the Image Classification system with two clients, User Client and Administrator Client. Through the Administrator Client GUI, among other functionalities, the administrator can create nodes (VMs) and list the resources available on the distributed system. Likewise, through the User Client GUI, users can upload data, encrypt and decrypt data, and send encrypted data to the cloud for processing. During the operation of the application, after uploading the data, the user will have a set of standard machine learning algorithms to choose from to process the data. Currently, the framework provides two algorithms, SVM and NN. Once selected, the distributed machine learning algorithm with the HE implementation will be run on the distributed system, which will then return the answer in encrypted form to be decrypted when needed.

VII. Experiments

In this section, we describe the experiments we performed to analyze the performance of the overall framework. The first part of the experiments focused on analyzing how the distributed system can improve the performance of the homomorphic encryption operations. The second part of the experiments looked at the trade-offs between computation overheads of cloud analytics and the accuracy of their estimations.

A. Experiments Setup

As described in the implementation section, the current version of the DFASC framework includes two HE libraries, PALISADE and SEAL. Each of these two libraries implements a number of HE schemes including BGV, CKKS, and The Brakerski-Fan-Vercauteren (BFV) [32]. For these experiments, we used the BGV scheme as it is the most matured HE scheme. In addition, we incorporated two machine learning algorithms, SVM and NN, using both of the HE libraries, PALISADE and SEAL. To analyze the feasibility of the framework, we implemented an image classification system,

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

which classifies images using both SVM and NN. We use this image classification system to perform the experiments.

For the experiments, we selected the *sklearn Digits Dataset* and used it as inputs for the two versions of the image classification tool, one for SVM and one for NN. The *sklearn Digits Dataset* is a popular standard dataset for classification written by scikit-learn [40]. It is made up of 1797 8x8 images, where each image is of a hand-written digit. In order to use this dataset in our experiments, we first transformed it into a feature vector with length 64, which defines its dimensionality. Overall, the dataset includes 10 classes, 180 samples per class with a total of 1797 images. The relevant features are integers from 0 to 16.

This dataset was uploaded into the Image Classification system through its GUI. We then encrypted the data homomorphically, and sent it to the distributed system for processing. For both algorithms, SVM and NN, we performed multiple runs by varying the dataset sizes and the numbers of nodes used in the distributed system.

B. Results of Experiments

Based on the above setup, we performed multiple experiments to analyze the performance of the DFASC framework in running the SVM and NN based analytic tools against the encrypted dataset. Specifically, we looked at the overhead incurred by the framework due to the expensive HE operations for both PALISADE and SEAL. During the experiment, the data was grouped into varying numbers of clusters as follows: 300, 600, 900, 1200, 1500, 1800. The distributed system was configured with varying numbers of nodes as follows: 1, 16, 32, 48, 64. Then, we ran the two analytic tools with each cluster size on each node configuration. Note that the times reported are execution times for the classifier and do not include the time taken to train the model.

1) Performance Analysis of the Distributed Framework: In this experiment, we analyzed the performance of the distributed framework as we increase the number of nodes for a given workload. The following four tables and figures summarize the results of the experiments.

Image Classification using SVM Implemented with PALISADE In this experiment, we ran the Image Classifier tool using SVM implemented with PALISADE. Table I shows the average running time of the classifier as we vary both the number of images and the number of nodes in the distributed system. The data from Table I is plotted in Figure 3. Note that, when running on a single node, the range of the running time is between 5 and 45 seconds for all six clusters of images. Figure 3 shows that, for a given cluster of images, the running time of the Image Classifier is decreasing exponentially as we increase the number of nodes in the distributed system (from 1 to 64). The largest gain in performance is when going from one node to thirty two nodes. Beyond thirty two nodes, the running time does not decrease further.

TABLE I: Image Classifier using SVM, Implemented with PALISADE (data in seconds)

Images	1	16	32	48	64	Total
300	7.57	1.30	1.20	1.20	1.60	12.87
600	15.10	1.80	1.80	1.90	2.10	22.70
900	22.78	2.30	2.00	2.50	2.80	32.38
1200	30.58	3.00	2.60	2.60	3.30	42.08
1500	37.57	3.80	3.40	3.10	3.60	51.47
1800	45.42	4.40	3.60	4.00	4.10	61.52
Total	159.02	16.60	14.60	15.30	17.50	223.02

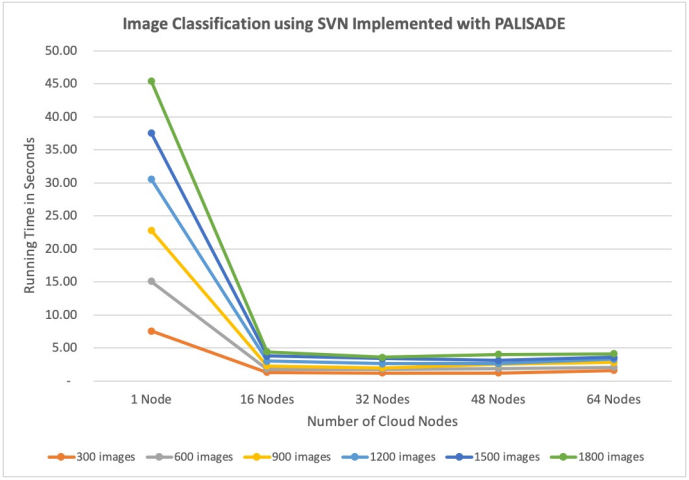


Fig. 3: Image Classifier using SVM, Implemented with PALISADE

Image Classification using NN Implemented with PALISADE In this experiment, we ran the Image Classifier tool using NN implemented with PALISADE. Table II shows the average running time of the classifier as we vary both the number of images and the number of nodes in the distributed system. The data from Table II is plotted in Figure 4. Note that, when running on a single node, the range of the running time is between 15 and 100 seconds for all six clusters of images. Figure 4 shows that, for a given cluster of images, the running time of the Image Classifier is decreasing exponentially as we increase the number of nodes in the distributed system (from 1 to 64).

Image Classification using SVM Implemented with SEAL In this experiment, we ran the Image Classifier tool using SVM implemented with SEAL. Table III shows the av-

TABLE II: Image Classifier using NN, Implemented with PALISADE (data in seconds)

Images	1	16	32	48	64	Total
300	16.08	2.10	1.60	1.70	1.70	23.18
600	31.87	3.00	2.70	2.70	3.10	43.37
900	47.95	3.90	2.80	3.60	3.60	61.85
1200	64.03	4.80	4.20	3.50	4.50	81.03
1500	79.80	6.30	5.30	4.60	5.50	101.50
1800	96.18	8.10	5.00	6.00	5.90	121.18
Total	335.92	28.20	21.60	22.10	24.30	432.12

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

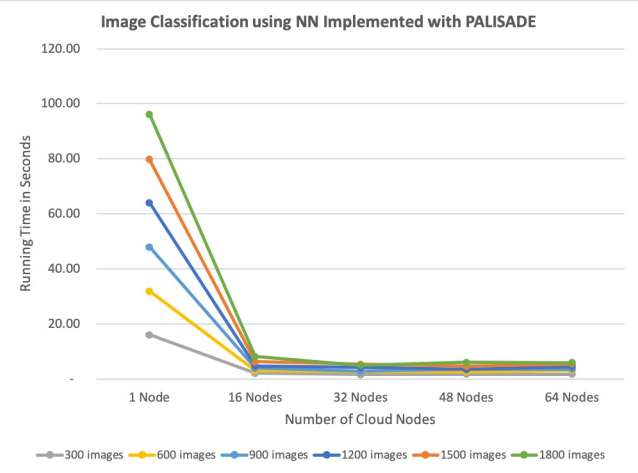


Fig. 4: Image Classifier using NN Implemented with PALISADE

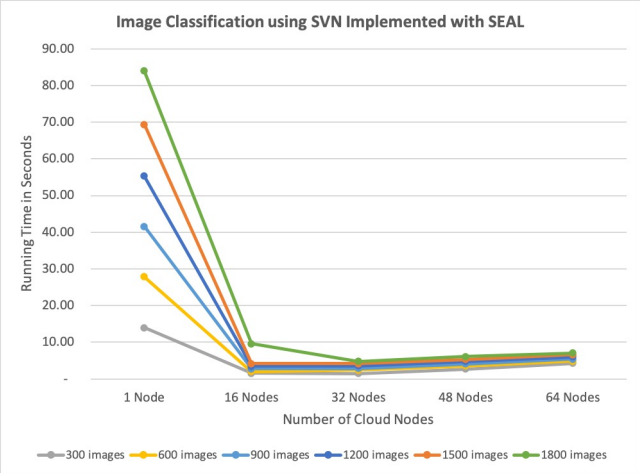


Fig. 5: Image Classifier using SVM, Implemented with SEAL

TABLE III: Image Classifier using SVM, Implemented with SEAL (data in seconds)

Images	1	16	32	48	64	Total
300	13.93	1.60	1.50	2.70	4.30	24.03
600	27.83	2.00	2.40	3.60	4.90	40.73
900	41.62	2.80	2.90	4.10	5.40	56.82
1200	55.30	3.50	3.60	4.80	6.00	73.20
1500	69.37	4.20	4.20	5.30	6.60	89.67
1800	83.98	9.60	4.80	6.10	7.10	111.58
Total	292.03	23.70	19.40	26.60	34.30	396.03

TABLE IV: Image Classifier using NN, Implemented with SEAL (data in seconds)

Images	1	16	32	48	64	Total
300	29.05	2.50	2.40	3.90	6.20	44.05
600	57.48	2.80	3.40	4.70	6.50	74.88
900	86.15	3.90	3.80	5.30	7.20	106.35
1200	115.22	4.60	4.90	5.80	8.00	138.52
1500	144.13	5.80	5.80	6.90	8.80	171.43
1800	172.47	11.00	6.20	7.40	8.90	205.97
Total	604.50	30.60	26.50	34.00	45.60	741.20

erage running time of the classifier as we vary both the number of images and the number of nodes in the distributed system. The data from Table III is plotted in Figure 5. Note that, when running on a single node, the range of the running time is between 15 and 85 seconds for all six clusters of images. Figure 5 shows that, for a given cluster of images, the running time of the Image Classifier is decreasing exponentially as we increase the number of nodes in the distributed system (from 1 to 64).

Image Classification using NN Implemented with SEAL
In this experiment, we ran the Image Classifier tool using NN implemented with SEAL. Table IV shows the average running time of the classifier as we vary both the number of images and the number of nodes in the distributed system. The data from Table IV is plotted in Figure 6. Note that, when running on a single node, the range of the running time is between 30 and 175 seconds for all six clusters of images. Figure 6 shows that, for a given cluster of images, the running time of the Image Classifier is decreasing exponentially as we increase the number of nodes in the distributed system (from 1 to 64).

For a given cluster of images, as the number of nodes is increased, the time taken to execute the classifier is reduced. However, the optimal value is between 32 and 48 nodes. As the number of nodes is increased beyond 48, the time taken increases again. This is true for both classification algorithms and both HE libraries. One possible explanation for this is that, since all of the nodes were implemented as virtual machines on one physical machine, the overhead associated with each running virtual machine competed with the speedup gained from parallelizing the algorithm across multiple virtual machines. As the number of virtual machines was increased and then exceeded the number of physical cores on the machine, significant context switching overhead

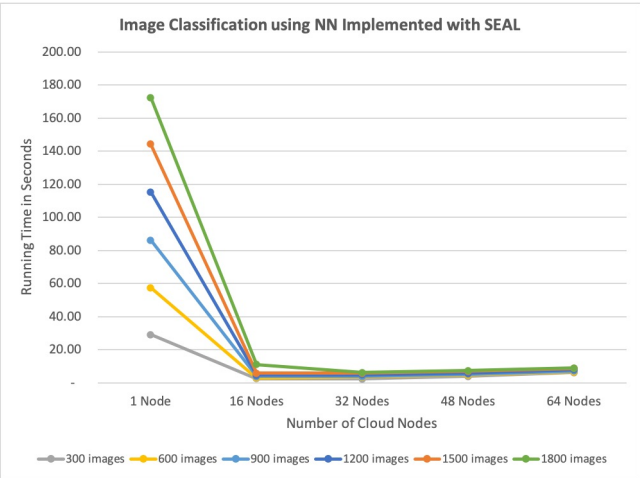


Fig. 6: Image Classifier using NN, Implemented with SEAL

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

was introduced, which countered any performance gains from increasing parallelism.

2) Trade-off Between Computation Overhead and Accuracy: In this experiment, we compared the performance of SVM and NN in classifying the images in the dataset. We used the combination of the selected six clusters of images (300, 600, 900, 1200, 1500, 1800) and five clusters of cloud nodes (1, 16, 32, 48, 64). This combination resulted in 30 different sets of data to be used as inputs for the two analytics. The results of running the two analytics against the 30 datasets are summarized in the previous four tables. Figure 7 shows a graph of a snapshot of these experiments, where the number of nodes is fixed to 16 and the number of datasets varied through all six clusters. As can be seen, the SVM based analytic ran exponentially faster than the NN based analytic. On the other side, the SVM based analytic was less accurate than the NN based analytic. The average accuracy after running all experiments was 85.44% for SVM based analytic and 94.65% for NN based analytic. From these results, we can observe that there is a tradeoff between the computation overhead and the accuracy of the analytics.

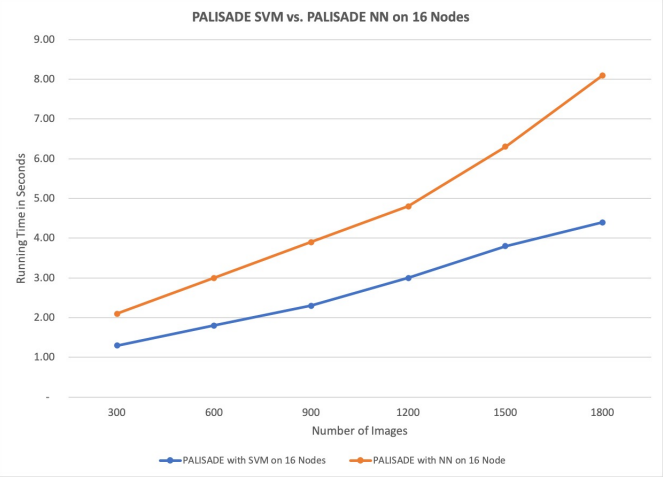


Fig. 7: SEAL and SVM based Image Classifier with 16 Nodes

Storage Overhead This experiment focused on analyzing the storage overhead associated with the ciphertext. The size of each image in the dataset is 780 bytes. After the image is encrypted homomorphically, the size of the cipher image expands to 29,367,027 bytes for PALISADE and 467,807 bytes for SEAL. The difference between PALISADE and SEAL resides in the techniques used by each library to encode the ciphertext.

3) Performance of Sharing: As described earlier, we use an email-based protocol for exchanging private keys to enable secure data sharing between users of the system. The email server provides an external channel to be used to securely share decryption keys. For this experiment we analyze the performance of the sharing protocol. The protocol includes exchanging public keys for sharing, sharing decryption keys, retrieving the shared encrypted data from the cloud, and decrypting the data. We executed this sharing protocol multiple times for both, SVM and NN, and computed the average execution time. We observed that the average time for both (0.90 seconds for SVM and 0.81 for NN) is less than a second. For data sharing purposes, this delay is not very noticeable to the user.

VIII. Related Work

Different techniques have been proposed for securing cloud analytics to increase their adoption [42]. These techniques in general use a combination of deterministic and randomized encryption, where a trade-off between the two is employed. Deterministic encryption supports a limited form of queries to be performed on the encrypted data, but is less secure, while randomized encryption is very secure, but does not support any operation on the encrypted data. Provably secure searchable encryption is an example of such techniques.

Provably secure searchable encryption (SE), including searchable symmetric encryption (SSE) and public key encryption with keyword search (PEKS), enables limited queries to be performed on encrypted data using encrypted keywords [43], [44]. SE techniques are in general expensive in terms of their computation and storage overhead. These techniques include trade-offs between security, efficiency, and functionality [45]. However, as observed in [43], regardless of SE schemes efficiency drawbacks, there is a noticeable lack of query expressiveness that hinders deployment in practice. With homomorphic encryption based security techniques, there is no limit on the number or type of operations that can be performed on the encrypted data.

Using HE to enable machine learning algorithms, including deep learning, to process data securely has gained attention in the research community in recent years [46], [47], [48], [49]. Many of the proposed approaches focus on using a given HE scheme to implement a specific machine learning algorithm. In [47], the authors show that it is possible to use a SHE scheme to implement a linear SVM to classify images for facial recognition. They extended Gentry's SHE scheme to work with low-degree polynomial functions, which are not limited by Hamming distance or linear projection. In [50], the authors went further by proposing an approach for implementing a non-linear SVM for classifying images in general using a SHE scheme. CryptoNets [51] uses the Microsoft SEAL HE library to implement deep learning algorithms. HE parallelization is limited to SIMD operations provided by the HE scheme. Faster CryptoNets [52] improves the performance of CryptoNets by leveraging the sparse representations throughout the neural network to optimize the HE operations and improve their performance. MSCryptoNet [53], based on multi-scheme FHE, protects the evaluation of the classifier, where the inputs can be encrypted with different encryption schemes and different keys. More information on current trends in using HE to process big data can be found in [54]. Unlike the above approaches, we are proposing a general framework for securing cloud analytics. We focus particularly on improving

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

the performance of analytic tools, implemented with HE, by distributing their executions in the cloud and providing a key management infrastructure for sharing the encrypted data.

HE schemes have also been considered as a means for securing statistical computations [55]. In [56], the authors demonstrate the feasibility of using HE in approximating conventional statistical regression methods. This approach takes advantage of the fact that estimation and prediction can both be performed in the encrypted domain; bootstrapping can be avoided even for moderately large problems; and scales linearly with the number of predictors. In [57], HE is used to develop a secure system that protects both the training and prediction data in logistic regression. Despite the non-linearity of both the training and prediction in logistic regression, this paper showed that it is feasible to use HE since only the addition operation is needed, which significantly improves performance compared to FHE. Our approach differs in that it provides a framework to enable developers to use a variety of analytic tools, which can be based on statistical analysis or other analytic techniques.

Privacy-preserving data splitting is another approach proposed to preserve data privacy in the cloud. In this approach, sensitive data is split in such a way that any partition by itself is not sensitive, and is stored separately. However, the techniques proposed are not very secure as they either don't support encryption or they support only limited operations to take place in the encrypted domain [58], [59]. Furthermore, these techniques are focusing more on preserving privacy of the data at rest rather than in processing.

Other proposed techniques for securing machine learning algorithms are based on multiparty computation (MPC) [60]. Fundamentally, MPC requires interactive communications among the different nodes to perform the computations, whereas our approach using HE allows computations to be performed independently by the nodes. In addition, since HE enables the computations to be performed without any key exchange, there is no overhead of secret sharing as in MPC. HE allows for empowering a single party to take advantage of the cloud to securely analyze and share their data with other parties.

IX. Conclusion and Future Work

In this paper, we propose DFASC, a distributed framework for secure computing in the cloud, to enable the development of secure distributed systems. Secure distributed systems developed using this framework allow for analytic tools to be implemented using HE and distributed throughout the nodes of the distributed system. These systems will provide a high level of data security for the analytic tools since data will remain encrypted during transit to and from the cloud, and during storage and processing in the cloud. In addition, the framework provides a simple but flexible technique for sharing encrypted data among users. This approach of using HE to provide data security during data processing addresses the shortcomings of standard cryptographic schemes such as the Advanced Encryption Standards and Blowfish, and addresses some of the vulnerabilities of outsourcing data to the cloud. This approach will enable organizations of all types and sizes to take advantage of large pools of computing resources available in the cloud without giving up the privacy of their data.

The challenge with the existing HE schemes resides in the computation and storage overheads they incur. We addressed the computation overhead by distributing the HE computations across multiple nodes to reduce the computation time. For future work, we plan on combining the high level distribution of HE libraries and the low level parallelization of the HE operations themselves proposed in the literature. For instance, one proposed technique is to use General-Purpose Graphics Processing Units (GPGPUs) to speed up the underlying operations of the HE libraries [61]. Combining these two approaches has the potential to significantly speed up the HE operations executed within the DFASC framework.

Currently, our framework includes two HE libraries, which both implement the BGV and CKKS HE scheme. To improve the validation of the framework, we plan to incorporate additional HE libraries with additional HE schemes into the framework. We will extend the framework to facilitate a trade-offs analysis of these libraries and schemes.

References

[1] M. Diallo, C. Graves, M. August, V. Rana, and K. Groarke, "DFSCC: A distributed framework for secure computation and sharing in the cloud," in Proceedings of the Sixth International Conference on Big Data, Small Data, Linked Data and Open Data (ALLDATA) 2020. IARIA XPS, 2020, pp. 27–33.

[2] A. B. Cummings, D. Eftekhary, and F. G. House, "Cloud computing market," Fortune Business Insights, 2019.

[3] S. Kumar, F. Morstatter, and H. Liu, Twitter data analytics. Springer, 2014.

[4] A. Alexandrov et al., "The stratosphere platform for big data analytics," The VLDB Journal—The International Journal on Very Large Data Bases, vol. 23, no. 6, 2014, pp. 939–964.

[5] F. Zulkernine et al., "Towards cloud-based analytics-as-a-service (claaas) for big data analytics in the cloud," in 2013 IEEE International Congress on Big Data. IEEE, 2013, pp. 62–69.

[6] D. Talia, "Clouds for scalable big data analytics," Computer, vol. 46, no. 5, 2013, pp. 98–101.

[7] S. K. Sharma and X. Wang, "Live data analytics with collaborative edge and cloud processing in wireless iot networks," IEEE Access, vol. 5, 2017, pp. 4621–4635.

[8] R. Ranjan, "Streaming big data processing in datacenter clouds," IEEE Cloud Computing, vol. 1, no. 1, 2014, pp. 78–83.

[9] J. L. Asenjo et al., "Industrial data analytics in a cloud platform," Sep. 6 2016, US Patent 9,438,648.

[10] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," IEEE Internet Computing, vol. 16, no. 1, 2012, pp. 69–73.

[11] D. Zissis and D. Lekkas, "Addressing cloud computing security issues," Future Generation computer systems, vol. 28, no. 3, 2012, pp. 583–592.

[12] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in 2009 Fifth International Joint Conference on INC, IMS and IDC. IEEE, 2009, pp. 44–51.

[13] H. Takabi, J. B. Joshi, and G.-J. Ahn, "Security and privacy challenges in cloud computing environments," IEEE Security & Privacy, vol. 8, no. 6, 2010, pp. 24–31.

[14] N. Gruschka and M. Jensen, "Attack surfaces: A taxonomy for attacks on cloud services," in 2010 IEEE 3rd international conference on cloud computing. IEEE, 2010, pp. 276–279.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[15] C. Gentry, "Fully homomorphic encryption using ideal lattices," in Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing, ser. STOC '09. New York, USA: ACM, 2009, pp. 169–178.

[16] A. Acar, H. Aksu, A. S. Uluagac, and M. Conti, "A survey on homomorphic encryption schemes: Theory and implementation," CoRR, vol. abs/1704.03578, 2017.

[17] M. H. Diallo, M. August, R. Hallman, M. Kline, H. Au, and V. Beach, "Callforfire: A mission-critical cloud-based application built using the nomad framework," in Financial Cryptography and Data Security - FC 2016 International Workshops, BITCOIN, VOTING, and WAHC, Christ Church, Barbados, February 26, 2016, Revised Selected Papers, ser. Lecture Notes in Computer Science, J. Clark, S. Meiklejohn, P. Y. A. Ryan, D. S. Wallach, M. Brenner, and K. Rohloff, Eds., vol. 9604. Springer, 2016, pp. 319–327. [Online]. Available: https://doi.org/10.1007/978-3-662-53357-4_21

[18] M. H. Diallo, B. Hore, E. Chang, S. Mehrotra, and N. Venkatasubramanian, "CloudProtect: Managing data privacy in cloud applications," in 2012 IEEE Fifth International Conference on Cloud Computing, June 2012, pp. 303–310.

[19] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. U. Khan, "The rise of "big data" on cloud computing: Review and open research issues," Information systems, vol. 47, 2015, pp. 98–115.

[20] V. N. Inukollu, S. Arsi, and S. R. Ravuri, "Security issues associated with big data in cloud computing," International Journal of Network Security & Its Applications, vol. 6, no. 3, 2014, p. 45.

[21] R. Toshniwal, K. G. Dastidar, and A. Nath, "Big data security issues and challenges," International Journal of Innovative Research in Advanced Engineering (IJIRAE), vol. 2, no. 2, 2015.

[22] Y. Gahi, M. Guennoun, and H. T. Mouftah, "Big data analytics: Security and privacy challenges," in 2016 IEEE Symposium on Computers and Communication (ISCC). IEEE, 2016, pp. 952–957.

[23] R. L. Rivest, L. Adleman, M. L. Dertouzos et al., "On data banks and privacy homomorphisms," Foundations of secure computation, vol. 4, no. 11, 1978, pp. 169–180.

[24] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in Annual International Conference on the Theory and Applications of Cryptographic Techniques. Springer, 2010, pp. 1–23.

[25] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in Proceedings of the 31st Annual Conference on Advances in Cryptology, ser. CRYPTO'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 505–524.

[26] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," ACM Transactions on Computation Theory (TOCT), vol. 6, no. 3, 2014, pp. 1–36.

[27] S. Halevi and V. Shoup, "Algorithms in helib," in Annual Cryptology Conference. Springer, 2014, pp. 554–571.

[28] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in International Conference on the Theory and Application of Cryptology and Information Security. Springer, 2017, pp. 409–437.

[29] S. S. Sathya, P. Vepakomma, R. Raskar, R. Ramachandra, and S. Bhattacharya, "A review of homomorphic encryption libraries for secure computation," arXiv preprint arXiv:1812.02428, 2018.

[30] K. Rohloff and G. Ryan, "The palisade lattice cryptography library," 2017, retrieved: 01, 2020.

[31] S. S. Sathya, P. Vepakomma, R. Raskar, R. Ramachandra, and S. Bhattacharya, "A review of homomorphic encryption libraries for secure computation," CoRR, vol. abs/1812.02428, 2018.

[32] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption." IACR Cryptol. ePrint Arch., vol. 2012, 2012, p. 144.

[33] G. Wang, Q. Liu, J. Wu, and M. Guo, "Hierarchical attribute-based encryption and scalable user revocation for sharing data in cloud servers," computers & security, vol. 30, no. 5, 2011, pp. 320–331.

[34] Q. Liu, G. Wang, and J. Wu, "Time-based proxy re-encryption scheme for secure data sharing in a cloud environment," Information sciences, vol. 258, 2014, pp. 355–370.

[35] C.-C. Lee, P.-S. Chung, and M.-S. Hwang, "A survey on attribute-based encryption schemes of access control in cloud environments." IJ Network Security, vol. 15, no. 4, 2013, pp. 231–240.

[36] A. Holovaty and J. Kaplan-Moss, The definitive guide to Django: Web development done right. Apress, 2009.

[37] Apache Software Foundation, "Apache hadoop," https://hadoop.apache.org, last accessed on 12/08/20.

[38] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin et al., "Apache spark: a unified engine for big data processing," Communications of the ACM, vol. 59, no. 11, 2016, pp. 56–65.

[39] J.-Y. Hwang, S.-B. Suh, S.-K. Heo, C.-J. Park, J.-M. Ryu, S.-Y. Park, and C.-R. Kim, "Xen on arm: System virtualization using xen hypervisor for arm-based secure mobile phones," in 5th IEEE Consumer Communications and Networking Conference. IEEE, 2008, pp. 257–261.

[40] D. Dua and C. Graff, "UCI Machine Learning Repository," University of California, Irvine, School of Information and Computer Sciences, http://archive.ics.uci.edu/ml, last accessed on 12/08/20.

[41] Q. Liu, X. Lu, F. Luo, S. Zhou, J. He, and K. Wang, "Securebp from homomorphic encryption," Security and Communication Networks, vol. 2020, 06 2020, pp. 1–9.

[42] S. Sobati Moghadam and A. Fayoumi, "Toward securing cloud-based data analytics: A discussion on current solutions and open issues," IEEE Access, vol. 7, 2019, pp. 45 632–45 650.

[43] C. Bösch, P. Hartel, W. Jonker, and A. Peter, "A survey of provably secure searchable encryption," ACM Computing Surveys (CSUR), vol. 47, no. 2, 2014, pp. 1–51.

[44] F. Han, J. Qin, and J. Hu, "Secure searches in the cloud: A survey," Future Generation Computer Systems, vol. 62, 2016, pp. 66–75.

[45] B. Hore, E.-C. Chang, M. H. Diallo, and S. Mehrotra, "Indexing encrypted documents for supporting efficient keyword search," in Workshop on Secure Data Management. Springer, 2012, pp. 93–110.

[46] T. Graepel, K. Lauter, and M. Naehrig, "Ml confidential: Machine learning on encrypted data," in International Conference on Information Security and Cryptology. Springer, 2012, pp. 1–21.

[47] J. R. Troncoso-Pastoriza, D. González-Jiménez, and F. Pérez-González, "Fully private noninteractive face verification," IEEE Transactions on Information Forensics and Security, vol. 8, no. 7, 2013, pp. 1101–1114.

[48] M. Kim, Y. Song, S. Wang, Y. Xia, and X. Jiang, "Secure logistic regression based on homomorphic encryption: Design and evaluation," JMIR medical informatics, vol. 6, no. 2, 2018, p. e19.

[49] Y. Aono, T. Hayashi, L. Wang, S. Moriai et al., "Privacy-preserving deep learning via additively homomorphic encryption," IEEE Transactions on Information Forensics and Security, vol. 13, no. 5, 2017, pp. 1333–1345.

[50] A. Barnett et al., "Image classification using non-linear support vector machines on encrypted data." IACR Cryptology ePrint Archive, vol. 2017, 2017, p. 857.

[51] R. Gilad-Bachrach et al., "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in International Conference on Machine Learning, 2016, pp. 201–210.

[52] E. Chou and Others, "Faster cryptonets: Leveraging sparsity for real-world encrypted inference," arXiv preprint arXiv:1811.09953, 2018.

[53] P. Li et al., "Multi-key privacy-preserving deep learning in cloud computing," Future Generation Computer Systems, vol. 74, 2017, pp. 76–85.

[54] R. A. Hallman, M. H. Diallo, M. A. August, and C. T. Graves, "Homomorphic encryption for secure computation on big data," in IoTBDS, 2018.

[55] L. J. Aslett, P. M. Esperança, and C. C. Holmes, "A review of homomorphic encryption and software tools for encrypted statistical machine learning," arXiv preprint arXiv:1508.06574, 2015.

[56] P. M. Esperança, L. J. Aslett, and C. C. Holmes, "Encrypted accelerated least squares regression," 2017.

[57] Y. Aono, T. Hayashi, L. Trieu Phong, and L. Wang, "Scalable and secure logistic regression via homomorphic encryption," in Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy, ser. CODASPY '16. New York, NY, USA: ACM, 2016, pp. 142–144.

[58] D. Sánchez and M. Batet, "Privacy-preserving data outsourcing in the cloud via semantic data splitting," Computer Communications, vol. 110, 2017, pp. 187–201.

[59] N. Kaaniche and M. Laurent, "Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms," Computer Communications, vol. 111, 2017, pp. 120–141.

[60] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," in Proceedings of the 55th Annual Design Automation Conference. ACM, 2018, p. 2.

[61] M. H. Diallo, M. August, R. Hallman, M. Kline, H. Au, and S. M. Slayback, "Nomad: a framework for ensuring data confidentiality in mission-critical cloud-based applications," Data Security in Cloud Computing, Security, 2017, pp. 19–44.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Data Sanitisation Protocols for the Privacy Funnel with Differential Privacy Guarantees

Milan Lopuhaä-Zwakenberg, Haochen Tong, and Boris Škorić
Department of Mathematics and Computer Science
Eindhoven University of Technology
Eindhoven, the Netherlands
email: {m.a.lopuhaa,b.skoric}@tue.nl, h.tong@student.tue.nl

***Abstract*—In the Open Data approach, governments and other public organisations want to share their datasets with the public, for accountability and to support participation. Data must be opened in such a way that individual privacy is safeguarded. The Privacy Funnel is a mathematical approach that produces a sanitised database that does not leak private data beyond a chosen threshold. The downsides to this approach are that it does not give worst-case privacy guarantees, and that finding optimal sanitisation protocols can be computationally prohibitive. These problems are tackled by using differential privacy metrics, and by considering local protocols that operate on one entry at a time. It is shown that under both the Local Differential Privacy and Local Information Privacy leakage metrics, one can efficiently obtain optimal protocols. Furthermore, Local Information Privacy is more closely aligned to the privacy requirements of the Privacy Funnel scenario, and optimal protocols satisfying Local Information Privacy are more efficiently computable. This paper also considers the scenario where each user has multiple attributes, for which a side-channel resistant privacy criterion is defined, and efficient methods to find protocols satisfying this criterion, while still offering good utility, are given. Finally, Conditional Reporting is introduced, an explicit LIP protocol that can be used when the optimal protocol is infeasible to compute. Experiments on real-world and synthetic data confirm the validity of these methods. The main output of this paper consists of methods to compute optimal privacy protocols, and explicit privacy protocols when the former are unfeasible computationally.**

***Keywords*—Privacy funnel; local differential privacy; information privacy; database sanitisation; complexity.**

I. Introduction

This paper is an extended version of [1]. Under the Open Data paradigm, governments and other public organisations want to share their collected data with the general public. This increases a government's transparency, and it also gives citizens and businesses the means to participate in decision-making, as well as using the data for their own purposes. However, while the released data should be as faithful to the raw data as possible, individual citizens' private data should not be compromised by such data publication.

Let $\mathcal{X}$ be a finite set. Consider a database $\vec{X} = (X_1, \ldots, X_n) \in \mathcal{X}^n$ owned by a data aggregator, containing a data item $X_i \in \mathcal{X}$ for each user i (For typical database settings, each user's data is a vector of attributes $X_i = (X_i^1, \ldots, X_i^m)$; this will be considered in more detail in Section VI). This data may not be considered sensitive by itself, but it might be correlated to a secret S_i. For instance, X_i might contain the age, sex, weight, skin colour, and average blood pressure of person i, while S_i is the presence of some medical condition. To publish the data in a privacy-preserving manner, the aggregator releases a sanitised database $\vec{Y} = (Y_1, \ldots, Y_n)$, obtained from applying a sanitisation mechanism $\mathcal{R}$ to $\vec{X}$. In this setting, *privacy* is considered to be the extent to which one is unable to infer information about the S_i from the sanitised database $\vec{Y}$. One way to formulate this is by measuring the privacy leakage as the mutual information $\mathrm{I}(\vec{S};\vec{Y})$, and utility as the mutual information $\mathrm{I}(\vec{X};\vec{Y})$. This leads to the *Privacy Funnel* (PF) problem:

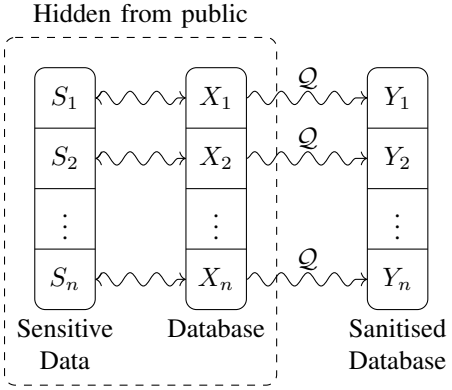


Figure 1. Model of PF with local protocols.

Problem 1. (Privacy Funnel, [2]) *Suppose the joint probability distribution of $\vec{S}$ and $\vec{X}$ is known to the aggregator, and let $M \in \mathbb{R}_{\geq 0}$. Then, find the sanitisation mechanism $\mathcal{R}$ such that $\mathrm{I}(\vec{X};\vec{Y})$ is maximised while $\mathrm{I}(\vec{S};\vec{Y}) \leq M$.*

There are two difficulties with this approach:

1) Finding and implementing good sanitisation mechanisms that operate on all of $\vec{X}$ can be computationally prohibitive for large n, as the complexity is exponential in n [3][4].
2) Taking mutual information as a leakage measure has as a disadvantage that it gives guarantees about the leakage in the average case. If n is large, this still leaves room for the sanitisation protocol to leak undesirably much information about a few unlucky users.

To deal with these two difficulties, two changes are made to the general approach. First, the focus is on *local* data sanitisation, i.e., optimisation protocols $\mathcal{Q}\colon \mathcal{X} \to \mathcal{Y}$ are

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

considered, for some finite set $\mathcal{Y}$, and $\mathcal{Q}$ is applied to each X_i individually; this situation is depicted in Figure 1. Local sanitisation can be implemented efficiently. In fact, this approach is often taken in the PF setting [5][3]. Second, to ensure strong privacy guarantees even in worst-case scenarios,stricter notions of privacy are considered, based on Local Differential Privacy (LDP) [6]. For these metrics, methods are developed to find optimal protocols. Furthermore, for situations where the optimal protocol is computationally unfeasible to find, a new protocol is introduced, *Conditional Reporting* (CR), that takes advantage of the fact that only S_i needs to be protected. Determining CR only requires finding the root of a onedimensional increasing function, which can be done fast numerically.

A. New contributions

In this paper, two Differential Privacy-like privacy metrics are adapted to the PF situation, namely ε-LDP [6] and Local Information Privacy (ε-LIP) [7][8]. These metrics are modified so that they measure leakage about the underlying S rather than X itself (for notational convenience, S, X, Y rather than S_i, X_i, Y_i is used throughout the rest of this paper). For a given level of leakage, the aim is to find the privacy protocol that maximises the mutual information between input X_i and output Y_i. Adapting methods from [9] on LDP and [10] on perfect privacy, the following Theorem is proven:

Theorem 1 (Theorems 2 and 3 paraphrased)**.** *Suppose X and S are discrete random variables on sets of size a and c, respectively. Suppose that their joint distribution and a privacy level $\varepsilon \geq 0$ are given.*

1) *The optimal ε-LDP protocol can be found by enumerating the vertices of a polytope in $a^2 - a$ dimensions defined by $a(c^2 - c)$ inequalities.*
2) *The optimal ε-LIP protocol can be found by enumerating the vertices of a polytope in $a - 1$ dimensions defined by $2ac$ inequalities.*

This theorem gives us methods to get data sanitisation protocols that give strong privacy guarantees, and optimal utility under these guarantees. This is important in settings where worst-case guarantees for privacy leakage are needed, rather than a bound on the average user's privacy.

Since the complexity of the polytope vertex enumeration depends significantly on both its dimension and the number of defining inequalities [11], finding optimal LIP protocols can be done significantly faster than finding optimal LDP protocols. Furthermore, it will be argued that LIP is a privacy metric that more accurately captures information leakage than LDP in the PF scenario. For these two reasons only LIP is considered in the remainder of the paper, although many results can also be formulated for LDP.

A common scenario is that a user's data X consists of multiple attributes, i.e., $X = (X^1, \ldots, X^m)$. Here one can consider an attacker model where the attacker has access to some of the X^j. In this situation ε-LIP does not accurately reflect a user's privacy. Because of this, a new privacy condition called *Side-channel Resistant LIP* is introduced that takes such sidechannels into account, and methods to find optimal protocols that satisfy this privacy condition are described.

Finding the optimal protocols can become computationally unfeasible for large a and c. In such a situation, one needs to resort to explicitely given protocols. In the literature there is a wealth of protocols that satisfy ε-LDP w.r.t. X. These certainly work in the PF situation, but they might not be ideal, because these are designed to obfuscate all information about X, rather than just the part that relates to S. For this reason, Conditional Reporting (CR) is introduced, a privacy protocol that focuses on hiding S rather than X. Finding the appropriate CR protocol for a given probability distribution and privacy level can be done fast numerically.

The structure of this paper is as follows. In Section II, an overview is given of related work on PF, LDP, and finding optimal protocols. The mathematical setting of this paper is formalised in Section III. In Sections IV and V, Theorem 1 is proven for LDP and LIP, respectively. In Section VI privacy in the multiple attribute scenario is discussed. Section VII is dedicated to Conditional Reporting and its privacy properties. In Section VIII, he methods and protocols discussed above are tested on both synthetic and real data. Compared to [1], new contents in this extended paper are Section VII, the experiments on real data, and the extended literature review.

II. Related work

The PF setting was introduced in [5], to provide a framework for obfuscating data in such a way that the obfuscated data remains as faithful as possible to the original, while ensuring that the information leakage about a latent variable is limited. PF is related to the Information Bottleneck (IB) [12], a problem from machine learning that seeks to compress data as much as possible, while retaining a minimal threshold of information about a latent variable. In PF as well as IB, both utility and leakage are measured via mutual information. Many approaches to finding the optimal protocols in PF also work for IB and vice versa [13][3]. A wider range of privacy metrics for PF, and their relation to Differential Privacy, is discussed in [8].

LDP was introduced in [6]. It is an adaptation of Differential Privacy (DP) [14] to a setting where there is no trusted central party to obfuscate the data. As a privacy metric, it has the advantage that it offers a privacy guarantee in any case, not just the average case, and that it does not depend on the data distribution. On the downside, it can be difficult to fulfill such a stringent definition of privacy, and many relaxations of (L)DP have been proposed [15][16][17][18]. Of particular interest to this paper is LIP [7][8], also called Removal Local Differential Privacy [19]. LIP retains the worst-case guarantees of LDP, but is less restrictive, and can take advantage of a known distribution. In the context where only part of the data is considered secret, many privacy metrics fall under the umbrella of Pufferfish Privacy [20].

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

In [9], a method was introduced for finding optimal LDP-protocols for a wide variety of utility metrics, including mutual information. The method relies on finding the vertices of a polytope, but since this is the well-studied Differential Privacy polytope, its vertices can be described explicitly [21]. Similarly, [10] uses a vertex enumeration method to find the optimal protocol in the perfect privacy situation, i.e., when the released data is independent of the secret data. The complexity of vertex enumeration is discussed in [22][11].

One can conclude that PF and LDP are both well-studied, and so are methods to find optimal LDP protocols. However, LDP-like metrics so far have not been applied to the PF scenario. The aim of this paper is to do so, and to find optimal PF protocols that satisfy LDP-like privacy requirements.

III. Mathematical Setting

The database $\vec{X} = (X_1, \ldots, X_n)$ consists of a data item X_i for each user i, each an element of a given finite set $\mathcal{X}$. Furthermore, each user has sensitive data $S_i \in \mathcal{S}$, which is correlated with X_i; again $\mathcal{S}$ is assumed to be finite (see Figure 1). Each (S_i, X_i) is assumed to be drawn independently from the same distribution $\mathrm{p}_{S,X}$ on $\mathcal{S} \times \mathcal{X}$ that is known to the aggregator through observing $(\vec{S}, \vec{X})$ (if one allows for non-independent X_i, then differential privacy is no longer an adequate privacy metric [15][8]). The aggregator, who has access to $\vec{X}$, sanitises the database by applying a sanitisation protocol (i.e., a random function) $\mathcal{Q}\colon \mathcal{X} \to \mathcal{Y}$ to each X_i, outputting $\vec{Y} = (Y_1, \ldots, Y_n) = (\mathcal{Q}(X_1), \ldots, \mathcal{Q}(X_n))$. The aggregator's goal is to find a $\mathcal{Q}$ that maximises the information about X_i preserved in Y_i (measured as $\mathrm{I}(X_i; Y_i)$) while leaking only minimal information about S_i.

Without loss of generality $\mathcal{X}, \mathcal{Y}, \mathcal{S}$ are identified with the sets $\{1, \ldots, a\}, \{1, \ldots, b\}, \{1, \ldots, c\}$, respectively, for integers a, b, c. The subscript i from X_i, Y_i, S_i is omitted as no probabilities depend on it, and probabilities are written as p_x, p_s, $\mathrm{p}_{x|s}$, etc., which form vectors p_X, $\mathrm{p}_{S|x}$, etc., and matrices $\mathrm{p}_{X|S}$, etc.

As noted before, instead of looking at the mutual information $\mathrm{I}(S;Y)$, two different, related measures of sensitive information leakage known from the literature are considered. The first one is an adaptation of LDP, the *de facto* standard in information privacy [6]:

Definition 1. (ε-LDP) *Let $\varepsilon \in \mathbb{R}_{\geq 0}$. say that $\mathcal{Q}$ satisfies ε-LDP w.r.t. S if*

$$\forall y \in \mathcal{Y}, \forall s, s' \in \mathcal{S}: \quad \frac{\mathbb{P}(Y = y | S = s)}{\mathbb{P}(Y = y | S = s')} \leq \mathrm{e}^{\varepsilon}. \tag{1}$$

Most literature on LDP considers LDP w.r.t. X, i.e., for all y, x, x' it holds that

$$\frac{\mathbb{P}(Y = y | X = x)}{\mathbb{P}(Y = y | X = x')} \leq \mathrm{e}^{\varepsilon}. \tag{2}$$

This is a stricter requirement, because under this definition all data needs to be protected, rather than just the underlying sensitive data. This typically comes at a cost in utility [10].

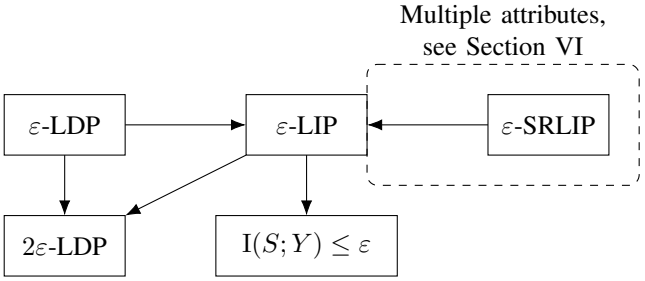


Figure 2. Relations between privacy notions. The multiple attributes setting is discussed in Section VI.

Throughout the present paper, ε-LDP always means ε-LDP w.r.t. S, unless otherwise specified.

The LDP metric reflects the fact that in the PF scenario one is only interested in hiding sensitive data, rather than all data; it is a specific case of what has been named Pufferfish Privacy [20]. The advantage of LDP compared to mutual information is that it gives privacy guarantees for the worst case, not just the average case. This is desirable in the database setting, as a worst-case metric guarantees the security of the private data of all users, while average-case metrics are only concerned with the average user. Another useful privacy metric is *Local Information Privacy* (LIP) [7][8], also called Removal Local Differential Privacy [19]:

Definition 2. (ε-LIP) *Let $\varepsilon \in \mathbb{R}_{\geq 0}$. The protocol $\mathcal{Q}$ satisfies ε-LIP w.r.t. S if*

$$\forall y \in \mathcal{Y}, s \in \mathcal{S}: \quad \mathrm{e}^{-\varepsilon} \leq \frac{\mathbb{P}(Y = y | S = s)}{\mathbb{P}(Y = y)} \leq \mathrm{e}^{\varepsilon}. \tag{3}$$

Compared to LDP, the disadvantage of LIP is that it depends on the distribution of S; this is not a problem in the PF scenario, as the aggregator, who chooses $\mathcal{Q}$, has access to the distribution of S. The advantage of LIP is that is more closely related to an attacker's capabilities: since

$$\frac{\mathbb{P}(Y = y | S = s)}{\mathbb{P}(Y = y)} = \frac{\mathbb{P}(S = s | Y = y)}{\mathbb{P}(S = s)}, \tag{4}$$

satisfying ε-LIP means that an attacker's posterior distribution of S given $Y = y$ does not deviate from their prior distribution by more than a factor e^{ε}. The following lemma outlines the relations between LDP, LIP and mutual information (see Figure 2).

Lemma 1. (See [8]) *Let $\mathcal{Q}$ be a sanitisation protocol, and let $\varepsilon \in \mathbb{R}_{\geq 0}$.*

1) *If $\mathcal{Q}$ satisfies ε-LDP, then it satisfies ε-LIP.*
2) *If $\mathcal{Q}$ satisfies ε-LIP, then it satisfies 2ε-LDP, and $\mathrm{I}(S;Y) \leq \varepsilon$.*

Remark 1. *One gets robust equivalents of LDP and LIP by demanding that $\mathcal{Q}$ satisfy ε-LIP (ε-LDP) for a set of distributions $\mathrm{p}_{S,X}$, instead of only a single distribution [20]. Letting $\mathrm{p}_{S,X}$ range over all possible distributions on $\mathcal{S} \times \mathcal{X}$ yields LIP (LDP) w.r.t. X.*

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

In this notation, instead of Problem 1 the following problem is considered:

Problem 2. *Suppose $\mathrm{p}_{S,X}$ is known to the aggregator, and let $\varepsilon \in \mathbb{R}_{\geq 0}$. Then, find the sanitisation protocol $\mathcal{Q}$ such that $\mathrm{I}(X;Y)$ is maximised while $\mathcal{Q}$ satisfies ε-LDP (ε-LIP, respectively) with respect to S.*

Note that this problem does not depend on the number of users n, and as such this approach will find solutions that are scalable w.r.t. n.

IV. Optimizing $\mathcal{Q}$ for ε-LDP

The goal is now to find the optimal $\mathcal{Q}$, i.e., the protocol that maximises $\mathrm{I}(X;Y)$ while satisfying ε-LDP, for a given ε. Any sanitisation protocol can be represented as a matrix $Q \in \mathbb{R}^{b\times a}$, where $Q_{y|x} = \mathbb{P}(Y = y|X = x)$. Then, ε-LDP is satisfied if and only if

$$\forall x\colon \ \sum_y Q_{y|x} = 1, \tag{5}$$

$$\forall x, y\colon \ 0 \leq Q_{y|x}, \tag{6}$$

$$\forall s, s', y\colon \ (Q\,\mathrm{p}_{X|s})_y \leq \mathrm{e}^{\varepsilon}(Q\,\mathrm{p}_{X|s'})_y. \tag{7}$$

As such, for a given $\mathcal{Y}$, the set of ε-LDP-satisfying sanitisation protocols can be considered a closed, bounded, convex polytope Γ in $\mathbb{R}^{b\times a}$. This fact allows us to efficiently find optimal protocols.

Theorem 2. *Let $\varepsilon \in \mathbb{R}_{\geq 0}$. Let $\mathcal{Q}\colon \mathcal{X} \to \mathcal{Y}$ be a ε-LDP protocol that maximises $\mathrm{I}(X;Y)$, i.e., the protocol that solves Problem 2 w.r.t. LDP.*

1) *One can take $b = a$.*
2) *Let Γ be the polytope described above, for $b = a$. Then the optimal $\mathcal{Q}$ corresponds to one of the vertices of Γ.*

Proof. The first result is obtained by generalising the results of [9]: there this is proven for regular ε-LDP (i.e., w.r.t. X), but the arguments given in that proof hold just as well in this situation; the only difference is that their polytope is defined by the ε-LDP conditions w.r.t. X, but this has no impact on the proof. The second statement follows from the fact that $\mathrm{I}(X;Y)$ is a convex function in $\mathcal{Q}$; therefore, its maximum on a bounded polytope is attained in one of the vertices. □

This theorem reduces the search for the optimal LDP protocol to enumerating the set of vertices of Γ, a $a(a-1)$-dimensional convex polytope. Note that the only property of $\mathrm{I}(X;Y)$ used in the proof is the fact that it is convex in Q. Therefore, the theorem holds for any convex utility metric.

One might argue that, since the optimal $\mathcal{Q}$ depends on $\mathrm{p}_{S,X}$, the publication of $\mathcal{Q}$ might provide an aggregator with information about the distribution of S. However, information on the distribution (as opposed to information of individual users' data) is not considered sensitive [23]. In fact, the reason why the aggregator sanitises the data is because an attacker is assumed to have knowledge about this correlation, and revealing too much information about X would cause the aggregator to use this information to infer information about S.

V. Optimizing $\mathcal{Q}$ for ε-LIP

If one uses ε-LIP as a privacy metric, one can find the optimal sanitisation protocol in a similar fashion. To do this, a sanitisation protocol $\mathcal{Q}$ is again described as a matrix, but this time a different one. Let $q \in \mathbb{R}^b$ be the probability mass function of Y, and let $R \in \mathbb{R}^{a\times b}$ be given by

$$R_{x|y} = \mathbb{P}(X = x|Y = y); \tag{8}$$

its y-th row is denoted by $R_{X|y} \in \mathbb{R}^a$. Then, a pair (R, q) defines a sanitisation protocol $\mathcal{Q}$ satisfying ε-LIP if and only if

$$\forall y\colon \ 0 \leq q_y, \tag{9}$$

$$Rq = \mathrm{p}_X, \tag{10}$$

$$\forall y\colon \ \sum_x R_{x|y} = 1, \tag{11}$$

$$\forall x, y\colon \ 0 \leq R_{x|y}, \tag{12}$$

$$\forall y, s\colon \ \mathrm{e}^{-\varepsilon}\,\mathrm{p}_s \leq \mathrm{p}_{s|X}\,R_{X|y} \leq \mathrm{e}^{\varepsilon}\,\mathrm{p}_s\,. \tag{13}$$

Note that (13) defines the ε-LIP condition, since for a given s, y one has

$$\frac{\mathrm{p}_{s|X}\,R_{X|y}}{\mathrm{p}_S} = \frac{\mathbb{P}(S = s|Y = y)}{\mathbb{P}(S = s)} = \frac{\mathbb{P}(Y = y|S = s)}{\mathbb{P}(Y = y)}. \tag{14}$$

(In)equalities (11–13) can be expressed as saying that for every $y \in \mathcal{Y}$ one has that $R_{X|y} \in \Delta$, where Δ is the convex closed bounded polytope in $\mathbb{R}^{\mathcal{X}}$ given by

$$\Delta = \left\{ v \in \mathbb{R}^{\mathcal{X}} : \begin{array}{l} \sum_x v_x = 1, \\ \forall x : 0 \leq v_x, \\ \forall s : \mathrm{e}^{-\varepsilon}\,\mathrm{p}_s \leq \mathrm{p}_{s|X}\,v \leq \mathrm{e}^{\varepsilon}\,\mathrm{p}_s \end{array} \right\}. \tag{15}$$

As in Theorem 2, this polytope can be used to find optimal protocols:

Theorem 3. *Let $\varepsilon \in \mathbb{R}_{\geq 0}$, and let Δ be the polytope above. Let $\mathcal{V} = \{v_1, \ldots, v_M\}$ be its set of vertices. For $v_i \in \mathcal{V}$, let $\mathrm{H}(v_i)$ be its entropy, i.e.*

$$\mathrm{H}(v_i) = -\sum_{x\in\mathcal{X}} v_{i,x} \ln(v_{i,x}). \tag{16}$$

Let $\hat{\alpha}$ be the solution to the optimisation problem

$$\begin{aligned} \text{minimise}_{\alpha\in\mathbb{R}^M} \quad & \sum_{i=1}^{M} \mathrm{H}(v_i)\alpha_i \\ \text{subject to} \quad & \forall i : \alpha_i \geq 0, \\ & \sum_{i=1}^{M} \alpha_i v_i = \mathrm{p}_X\,. \end{aligned} \tag{17}$$

Then the ε-LIP protocol $\mathcal{Q}\colon \mathcal{X} \to \mathcal{Y}$ that maximises $\mathrm{I}(X;Y)$ is given by

$$\mathcal{Y} = \{i \leq M : \hat{\alpha}_i > 0\}, \tag{18}$$

$$q_i = \hat{\alpha}_i, \tag{19}$$

$$R_{x|i} = v_{i,x}, \tag{20}$$

for all $i \in \mathcal{Y} \subseteq \{1, \ldots, M\}$ and all $x \in \mathcal{X}$. One has $b \leq a$.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Proof. This was proven for $\varepsilon = 0$ (i.e., when S and Y are independent) in [10], but the proof works similarly for $\varepsilon > 0$; the main difference is that the equality constraints of their (10) will be replaced by the inequality constraints of this paper's (13), but this has no impact on the proof presented there. □

Since linear optimisation problems can be solved fast, again the optimisation problem reduces to finding the vertices of a polytope. The advantage of using LIP instead of LDP is that Δ is a $(a-1)$-dimensional polytope, while Γ of Section IV is $a(a-1)$-dimensional. The time complexity of vertex enumeration is linear in the number of vertices [22], while the number of vertices can grow exponentially in the dimension of the polyhedron [11]. Together, this means that the dimension plays a huge role in the time complexity, hence the optimum under LIP is expected to be found significantly faster than under LDP.

VI. Multiple Attributes

An often-occuring scenario is that a user's data consists of multiple attributes, i.e.,

$$X = (X^1, \ldots, X^m) \in \mathcal{X} = \mathcal{X}^1 \times \cdots \times \mathcal{X}^m. \tag{21}$$

This can be problematic for this paper's approach for two reasons:

1) Such a large $\mathcal{X}$ can be problematic, since the computing time for optimisation both under LDP and LIP will depend heavily on a.
2) In practice, an attacker might sometimes utilise side channels to access some subsets of attributes X_i^j for some users. For these users, a sanitisation protocol can leak more information (w.r.t. to the attacker's updated prior information) than its LDP/LIP parameter would suggest.

To see how the second problem might arise in practice, suppose that X_i^1 is the height of individual i, X_i^2 is their weight, and S_i is whether i is obese or not. Since height is only lightly correlated with obesity, taking $Y_i = X_i^1$ would satisfy ε-LIP for some reasonably small ε. However, suppose that an attacker has access to X_i^2 via a side channel. While knowing i's weight gives the attacker some, but not perfect knowledge about i's obesity, the combination of the weight from the side channel, and the height from the Y_i, allows the attacker to calculate i's BMI, giving much more information about i's obesity. Therefore, the given protocol gives much less privacy in the presence of this side channel.

To solve the second problem, a more stringent privacy notion called *Side-channel Resistant LIP* (SRLIP) is introduced, which ensures that no matter which attributes an attacker has access to, the protocol still satisfies ε-LIP with respect to the attacker's new prior distribution. One could similarly introduce SRLDP, and many results will still hold for this privacy measure; nevertheless, since it has been concluded that LIP is preferable to LDP, the focus is on SRLIP. For any subset $J \subseteq \{1, \ldots, m\}$, the notation $\mathcal{X}^J$ is used for the set $\prod_{j \in J} \mathcal{X}^j$, and its elements are written as x^J.

Definition 3. (ε-SRLIP). *Let $\varepsilon > 0$, and let $\mathcal{X} = \prod_{j=1}^m \mathcal{X}^j$. The protocol $\mathcal{Q}$ satisfies ε-SRLIP if for every $y \in \mathcal{Y}$, for every $s \in \mathcal{S}$, for every $J \subseteq \{1, \ldots, m\}$, and for every $x^J \in \mathcal{X}^J$ one has*

$$\mathrm{e}^{-\varepsilon} \leq \frac{\mathbb{P}(Y = y | S = s, X^J = x^J)}{\mathbb{P}(Y = y | X^J = x^J)} \leq \mathrm{e}^{\varepsilon}. \tag{22}$$

In terms of Remark 1, $\mathcal{Q}$ satisfies ε-SRLIP if and only if it satisfies ε-LIP w.r.t. $\mathrm{p}_{S,X|x^J}$ for all J and x^J. Taking $J = \varnothing$ gives us the regular definition of ε-LIP, proving the following Lemma:

Lemma 2. *Let $\varepsilon > 0$. If $\mathcal{Q}$ satisfies ε-SRLIP, then $\mathcal{Q}$ satisfies ε-LIP.*

While SRLIP is stricter than LIP itself, it has the advantage that even when an attacker has access to some data of a user, the sanitisation protocol still does not leak an unwanted amount of information beyond the knowledge the attacker has gained via the side channel. Another advantage is that, contrary to LIP itself, SRLIP satisfies an analogon of the concept of *privacy budget* [14]:

Theorem 4. *Let $\mathcal{X} = \prod_{j=1}^m \mathcal{X}^j$, and for every j, let $\mathcal{Q}^j \colon \mathcal{X}^j \to \mathcal{Y}^j$ be a sanitisation protocol. Let $\varepsilon^j \in \mathbb{R}_{\geq 0}$ for every j. Suppose that for every $j \leq m$, for every $J \subseteq \{1, \ldots, j-1, j+1, \ldots, m\}$, and every $x^J \in \mathcal{X}^J$, $\mathcal{Q}^j$ satisfies ε^j-LIP w.r.t. $\mathrm{p}_{S,X|x^J}$. Then $\prod_j \mathcal{Q}^j \colon \mathcal{X} \to \prod_j \mathcal{Y}^j$ satisfies $\sum_j \varepsilon^j$-SRLIP.*

The proof is presented in Appendix A. This theorem tells us that to find a ε-SRLIP protocol for $\mathcal{X}$, it suffices to find a sanitisation protocol for each $\mathcal{X}^j$ that is $\frac{\varepsilon}{m}$-LIP w.r.t. a number of prior distributions. Unfortunately, the method of finding an optimal ε-LIP protocol w.r.t. one prior $\mathrm{p}_{S,X}$ of Theorem 3 does not transfer to the multiple prior setting. This is because this method only finds one (R, q), while by (10) a different (R, q) is needed for each prior distribution. Therefore, an approach similar to the one in Theorem 2 is adopted. The matrix Q^j (given by $Q^j_{y^j|x^j} = \mathbb{P}(\mathcal{Q}^j(x^j) = y^j)$) corresponding to $\mathcal{Q}^j \colon \mathcal{X}^j \to \mathcal{Y}^j$ satisfies the criteria of Theorem 4 if and only if the following criteria are satisfied:

$$\forall x^j : \sum_{y^j} Q^j_{y^j|x^j} = 1, \tag{23}$$

$$\forall x^j, y^j : 0 \leq Q^j_{y^j|x^j}, \tag{24}$$

$$\forall J, x^J, s, y^j : \mathrm{e}^{-\varepsilon/m} (Q^j \, \mathrm{p}_{X^j|x^J})_{y^j} \leq (Q^j \, \mathrm{p}_{X^j|s,x^J})_{y^j}, \tag{25}$$

$$\forall J, x^J, s, y^j : (Q^j \, \mathrm{p}_{X^j|s,x^J})_{y^j} \leq \mathrm{e}^{\varepsilon/m} (Q^j \, \mathrm{p}_{X^j|x^J})_{y^j}. \tag{26}$$

Similar to Theorem 2, the optimal $\mathcal{Q}^j$ satisfying these conditions can be found by finding the vertices of the polytope defined by (23–26). In terms of time complexity, the comparison to finding the optimal ε-LIP protocol via Theorem 3 versus finding a ε-SRLIP protocol via Theorem 4 is not straightforward. The complexity of enumerating the vertices of a polytope is $\mathcal{O}(ndv)$, where n is the number of inequalities, d is the dimension, and v is the number of vertices [22]. For the

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

Δ of Theorem 3 one has $d = a-1$ and $n = a+2c$. By contrast, the polytope defined by (23–26) satisfies $d = a^j(a^j-1)$ and $n = (a^j)^2 + 2c\prod_{j'\neq j}(a^{j'}+1)$. Finding v for both these polytopes is difficult, but in general $v \leq \binom{n}{d}$. Since this grows exponentially in d, Theorem 4 is expected to be faster when the a^j are small compared to a, i.e., when m is large. This will be investigated experimentally in Section VIII.

VII. Explicit protocols

The methods of Sections IV and V allow us to find the optimal LDP and LIP protocols. The complexity depends heavily on a and c, and can become computationally infeasible for large a and c. For such datasets, one has to rely on predetermined privacy algorithms. Two approaches are introduced: as a benchmark, Section VII-A discusses how 'standard' LDP protocols can be applied to the PF situation, and a new method, Conditional Reporting, that is meant to address the shortcomings of standard LDP protocols, is introduced in Section VII-B. As in the previous section, the focus is on LIP, but much of the discussion carries over to LDP as well.

A. Standard LDP protocols

In the literature, there are many examples of protocols $\mathcal{Q}\colon \mathcal{X} \to \mathcal{Y}$, depending on a privacy parameter α, whose output satisfies α-LDP with respect to X; for an overview see [24]. Such a protocol automatically satisfies α-LDP, hence certainly α-LIP, with respect to S. However, because X is only indirectly correlated with Y, such a protocol's actual LIP value may be better. The privacy of such a protocol $\mathcal{Q}$ is found by

$$\mathrm{LIP}(\mathcal{Q}) = \max_{y\in\mathcal{Y},s\in\mathcal{S}} \left| \ln \frac{\sum_x Q_{y|x}\,\mathrm{p}_{x|s}}{\sum_x Q_{y|x}\,\mathrm{p}_x} \right|; \tag{27}$$

then $\mathcal{Q}$ satisfies ε-LIP if and only if $\mathrm{LIP}(\mathcal{Q}) \leq \varepsilon$.

This paper considers two LDP protocols. The first one is Generalised Rapid Response (GRR) [25]. The key strength of GRR is that for large enough α it maximises $\mathrm{I}(X;Y)$ [9]. Given α, GRR is a privacy protocol $\mathrm{GRR}^\alpha\colon \mathcal{X} \to \mathcal{X}$ given by

$$\mathrm{GRR}^\alpha_{y|x} = \begin{cases} \frac{\mathrm{e}^\alpha}{\mathrm{e}^\alpha+a-1}, & \text{if } x = y, \\ \frac{1}{\mathrm{e}^\alpha+a-1}, & \text{if } x \neq y. \end{cases} \tag{28}$$

A direct calculation then shows that

$$\mathrm{LIP}(\mathrm{GRR}^\alpha) = \max_{x,s} \left| \ln \frac{1+(\mathrm{e}^\alpha-1)\,\mathrm{p}_{x|s}}{1+(\mathrm{e}^\alpha-1)\,\mathrm{p}_x} \right|. \tag{29}$$

For GRR to satisfy ε-LIP, the equation $\mathrm{LIP}(\mathrm{GRR}^\alpha) = \varepsilon$ needs to be solved for α. Since $\mathrm{LIP}(\mathrm{GRR}^\alpha)$ is increasing in α, this can be done fast computationally.

The second protocol that is relevant to this paper is Optimised Unary Encoding (OUE) [26]. This protocol is notable for being one of the protocols that has the least known variance in frequency estimation [26]. For a choice of α as privacy parameter, and an input x, the output of $\mathrm{OUE}^\alpha\colon \mathcal{X} \to 2^{\mathcal{X}}$ is a vector of independent Bernoulli variables $E_{x'}$ for $x' \in \mathcal{X}$, satisfying

$$\mathbb{P}(E_{x'} = 1) = \begin{cases} \frac{1}{2}, & \text{if } x' = x, \\ \frac{1}{\mathrm{e}^\alpha+1}, & \text{if } x' \neq x. \end{cases} \tag{30}$$

In other words, If a $y \in 2^{\mathcal{X}}$ is identified with a subset of $\mathcal{X}$ (so $\#y$ denotes its cardinality), one gets

$$\mathrm{OUE}^\alpha_{y|x} = \begin{cases} \frac{\mathrm{e}^{(a-\#y)\alpha}}{2(\mathrm{e}^\alpha+1)^{a-1}}, & \text{if } x \in y, \\ \frac{\mathrm{e}^{(a-\#y-1)\alpha}}{2(\mathrm{e}^\alpha+1)^{a-1}}, & \text{if } x \notin y. \end{cases} \tag{31}$$

It follows that

$$\mathrm{LIP}(\mathrm{OUE}^\alpha) = \max_{y,s} \left| \ln \frac{1+(\mathrm{e}^\alpha-1)\sum_{x\in y}\mathrm{p}_{x|s}}{1+(\mathrm{e}^\alpha-1)\sum_{x\in y}\mathrm{p}_x} \right|. \tag{32}$$

B. Conditional Reporting

In general, a generic LDP protocol will not be ideal for the PF scenario, since these are designed to obscure all information about X, rather than just the part that holds information about S. To address this shortcoming, the protocol *Conditional Reporting* (CR) is introduced in Algorithm 1. This mechanism needs both S and X as input; hence it differs from the other protocols discussed in this paper, which only have X as input. The value of S is masked by Randomised Response. If the output $\tilde{s}$ equals S, the algorithm returns the true value of X. If not, it outputs a random one, whose probability distribution is given by $\mathrm{p}_{X|\tilde{s}}$.

Algorithm 1: Conditional Reporting (CR^α)

Input : Privacy parameter α; Probability distribution $\mathrm{p}_{S,X}$; input $(s,x) \in \mathcal{S}\times\mathcal{X}$
Output: $y \in \mathcal{X}$

Sample $\tilde{s} \in \mathcal{S}$ with
$\mathbb{P}(\tilde{s} = s') = \begin{cases} \frac{\mathrm{e}^\alpha}{\mathrm{e}^\alpha+\#\mathcal{S}-1}, & \text{if } s' = s, \\ \frac{1}{\mathrm{e}^\alpha+\#\mathcal{S}-1}, & \text{otherwise} \end{cases}$
if $\tilde{s} = s$ **then**
  $y \leftarrow x$;
else
  Sample $\tilde{x} \in \mathcal{X}$ with $\mathbb{P}(\tilde{x} = x') = \mathrm{p}_{x'|\tilde{s}}$;
  $y \leftarrow \tilde{x}$;
end

CR^α certainly satisfies α-LDP, hence α-LIP, w.r.t. S. However, if S and X are not perfectly correlated, better privacy can be achieved, as outlined by the proposition below.

Proposition 1. *Given a probability distribution* $\mathrm{p}_{X,S}$ *and a* $\alpha \geq 0$*, define*

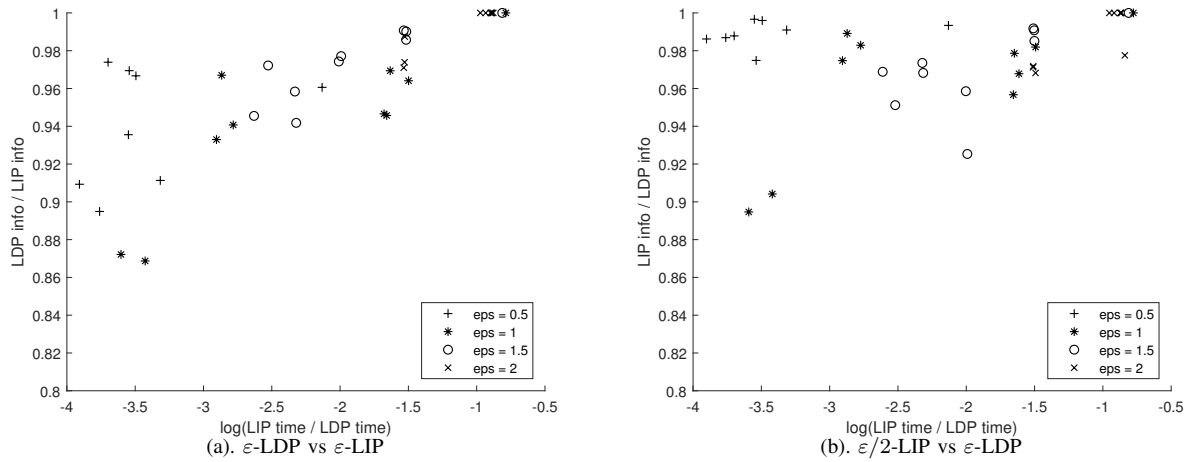
$$L(\alpha) = \max_{x,s} \left| \ln \frac{(\mathrm{e}^\alpha-1)\,\mathrm{p}_{x|s} + \sum_{s'}\mathrm{p}_{x|s'}}{(\mathrm{e}^\alpha-1)\,\mathrm{p}_x + \sum_{s'}\mathrm{p}_{x|s'}} \right|. \tag{33}$$

Then CR^α *satisfies* ε*-LIP if and only if* $\varepsilon \geq L(\alpha)$*.*

The proof is presented in Appendix A. One can use this proposition to find the α needed to have CR^α satisfy ε-LDP, by solving $L(\alpha) = \varepsilon$. At the very least one has the following upper bound:

Proposition 2. *The protocol* CR^α *satisfies* α*-LDP. In particular, it satisfies* α*-LIP, and* $L(\alpha) \leq \alpha$*.*

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(a). ε-LDP vs ε-LIP (b). $\varepsilon/2$-LIP vs ε-LDP

Figure 3. Comparison of computation time and $\mathrm{I}(X;Y)$ for LDP protocols found via Theorem 2 and LIP protocols found via Theorem 3, for random $\mathrm{p}_{S,X}$ with $c = 2$, $a = 5$, and $\varepsilon \in \{0.5, 1, 1.5, 2\}$.

Proof. For all $y \in \mathcal{X}$ and $s \in \mathcal{S}$ one has, following equation (48) in Appendix A, that

$$\mathbb{P}(\mathrm{CR}^{\alpha}(X,S) = y | S = s) = \tfrac{1}{\mathrm{e}^{\alpha}+c-1}\Big(\mathrm{e}^{\alpha}\,\mathrm{p}_{y|s} + \sum_{s' \neq s} \mathrm{p}_{y|s'}\Big). \tag{34}$$

It follows that

$$\begin{aligned}
&\frac{\mathbb{P}(\mathrm{CR}^{\alpha}(X,S) = y | S = s)}{\mathbb{P}(\mathrm{CR}^{\alpha}(X,S) = y | S = s')} \\
&= \frac{\mathrm{e}^{\alpha}\,\mathrm{p}_{y|s} + p_{y|s'} + \sum_{s'' \neq s,s'} \mathrm{p}_{y|s''}}{\mathrm{p}_{y|s} + \mathrm{e}^{\alpha} p_{y|s'} + \sum_{s'' \neq s,s'} \mathrm{p}_{y|s''}} && (35) \\
&\leq \max\left\{1, \frac{\mathrm{e}^{\alpha}\,\mathrm{p}_{y|s} + p_{y|s'}}{\mathrm{p}_{y|s} + \mathrm{e}^{\alpha} p_{y|s'}}\right\} && (36) \\
&\leq \mathrm{e}^{\alpha}. && \square
\end{aligned}$$

VIII. Experiments

The feasibility of the different methods is tested by performing small-scale experiments on synthetic data and real-world data. All experiments are implemented in Matlab and conducted on a PC with Intel Core i7-7700HQ 2.8GHz and 32GB memory.

A. Synthetic data: LDP vs LIP

The computing time for finding optimal ε-LDP and ε-LIP protocols was compared for $c = 2$ and $a = 5$ for 10 random distributions $\mathrm{p}_{S,X}$, obtained by generating each $\mathrm{p}_{s,x}$ uniformly from $[0, 1]$ and then normalising. The LDP/LIP privacy parameter ε is taken to be in $\{0.5, 1, 1.5, 2\}$; the results are in Figure 3(a). As one can see, Theorem 3 gives significantly faster results than Theorem 2; the average computing time for Theorem 2 for $\varepsilon = 0.5$ is 133s, while for Theorem 3 this is 0.0206s. With regards to the utility $\mathrm{I}(X;Y)$, since ε-LDP implies ε-LIP, the optimal ε-LIP protocol will have better utility than the optimal ε-LDP protocol. However, as can be seen from the figure, the difference in utility is relatively low.

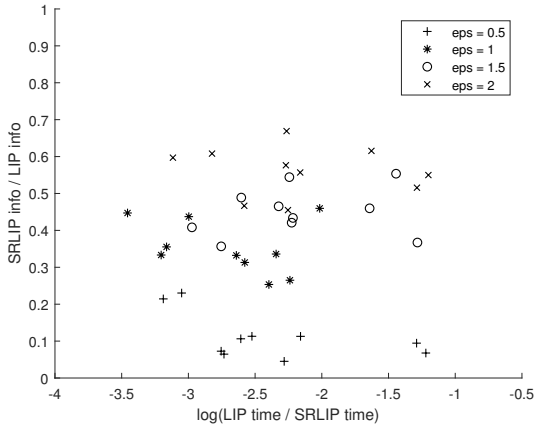


Figure 4. Comparison of computation time and $\mathrm{I}(X;Y)$ for ε-(SR)LIP-protocols found via Theorems 3 and 4, for random $\mathrm{p}_{S,X}$ with $c = 2$, $a_1 = a_2 = 3$, $a_3 = 4$, and $\varepsilon \in \{0.5, 1, 1.5, 2\}$.

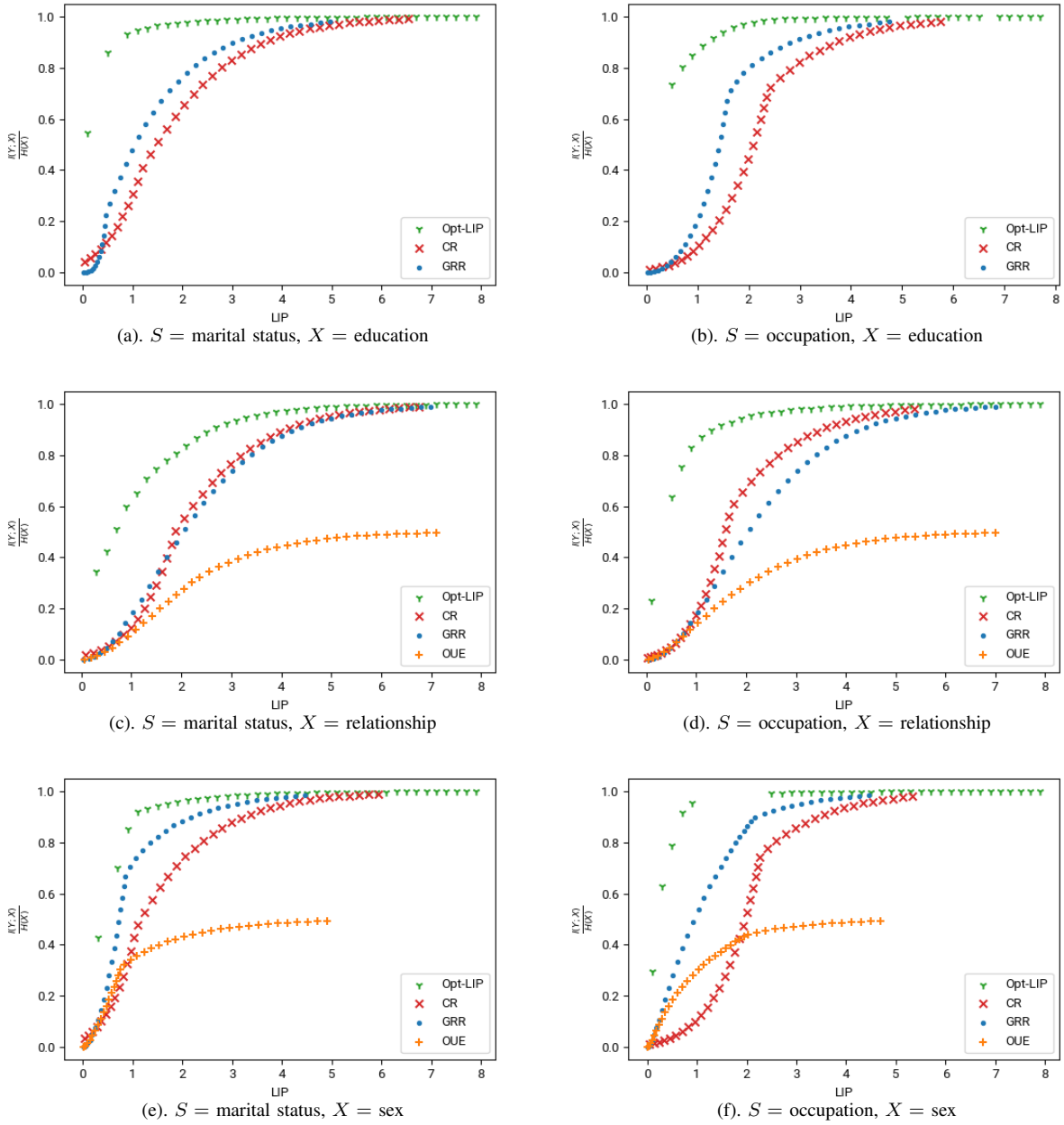
Note that for bigger ε, both the difference in computing time and the difference in $\mathrm{I}(X;Y)$ between LDP and LIP become less. This is because of the probabilistic relation between S and X, for ε large enough, any sanitisation protocol satisfies ε-LIP and ε-LDP. This means that as ε grows, the resulting polytopes will have fewer defining inequalities, hence they will have fewer vertices. This results in lower computation times, which affects LDP more than LIP. At the same time, the fact that every protocol is both ε-LIP and ε-LDP will result in the same optimal utility.

In Figure 3(b), optimal $\frac{\varepsilon}{2}$-LDP protocols are compared to to optimal ε-LIP protocols. Again, LIP is significantly faster than LDP. Since ε-LIP implies $\frac{\varepsilon}{2}$-LDP, the optimal $\frac{\varepsilon}{2}$-LDP has higher utility; again the difference is low.

B. Synthetic data: LIP vs SRLIP

Similar comparisons are perfomed for multiple attributes, for $c = 2$, $a_1 = a_2 = 3$ and $a_3 = 4$, comparing the methods

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(a). S = marital status, X = education

(b). S = occupation, X = education

(c). S = marital status, X = relationship

(d). S = occupation, X = relationship

(e). S = marital status, X = sex

(f). S = occupation, X = sex

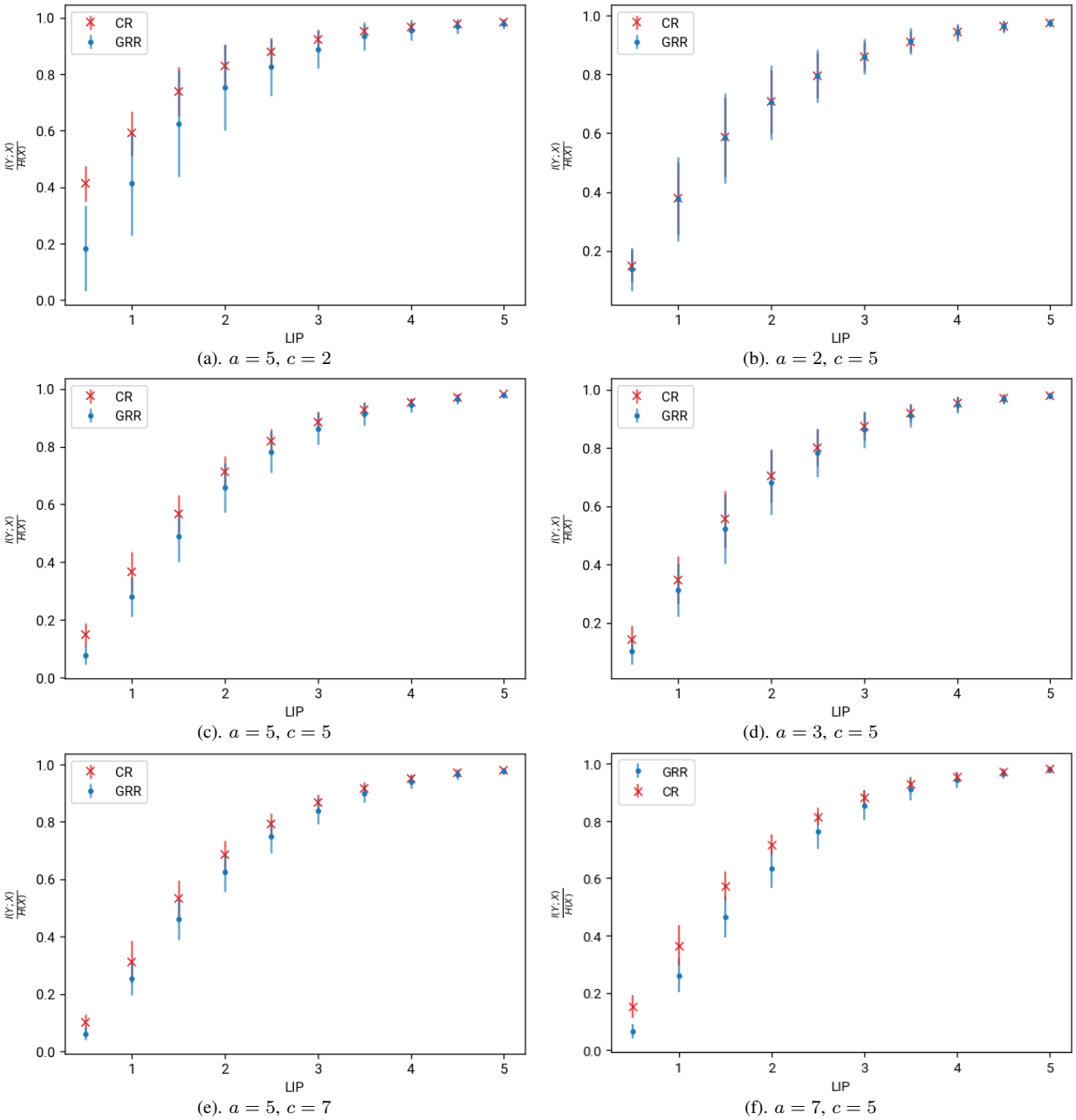
Figure 5. Experiments on the adult-dataset.

of Theorems 3 and 4. The results are presented in Figure 4. As one can see, Theorem 4 is significantly slower, with Theorem 3 being on average 476 times as fast. There is a sizable difference in utility, caused on one hand by the fact that ε-SRLIP is a stricter privacy requirement than ε-LIP, and on the other hand by the fact that Theorem 4 does not give us the optimal ε-SRLIP protocol.

C. Adult-dataset

The utility of Conditional Reporting (CR) is tested both on real world data and synthetic data. The real world data is from the well-known adult-dataset [27], which contains demographic data from the 1994 US census. For these experiments S is taken to be in $\{$marital status, occupation$\}$ (with $c = 7$ and $c = 15$, respectively) and X is taken to be in $\{$education, relationship, sex$\}$ (with $a = 16, 6, 2$). Based on the findings in the previous sections, LIP is taken as a privacy measure, and $\mathrm{I}(X;Y)$ as a utility measure. CR is compared on the one hand with the optimal method (Opt-LIP) found in Section V, and on the other hand with the established LDP protocols GRR and OUE. The results are shown in Figure 5. For X = education, the mutual information for OUE was infeasible to compute. Similarly, for S = occupation, some cases of Opt-LIP failed to compute within a reasonable

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(a). $a = 5$, $c = 2$ (b). $a = 2$, $c = 5$

(c). $a = 5$, $c = 5$ (d). $a = 3$, $c = 5$

(e). $a = 5$, $c = 7$ (f). $a = 7$, $c = 5$

Figure 6. Experiments on synthetic data. For each value of a and c, the average utility is taken over 100 randomly generated probability distributions. Bar size denotes standard deviation.

timeframe. Nevertheless, it can concluded that GRR and CR both perform somewhere between Opt-LIP and OUE. As the LIP value ε grows larger, GRR and CR grow close to Opt-LIP. At the same time, OUE falls off for large ε, having $\frac{1}{2}\,\mathrm{H}(X)$ as its limit. This is because OUE by design only has probability $\frac{1}{2}$ transmitting the true X (as element of the set Y). The difference between GRR and CR is less clear, and it appears to depend on the joint distribution $\mathrm{p}_{X,S}$ which protocol gives the best utility.

D. Synthetic data: GRR vs CR

To investigate the difference between GRR and CR, both methods are applied to synthetic data. OUE is disregarded as it performs worse than the other two protocols, especially in the low privacy regime. For a fixed choice of a and c, 100 probability distributions are drawn from the Jeffreys prior on $\mathcal{S}\times\mathcal{X}$, i.e., the symmetric Dirichlet distribution with parameter $\frac{1}{2}$. A set of LIP values ε is fixed, and for each of these and each probability distribution, equations (29) and (33) are solved, setting the left hand side equal to ε and solving for α_{GRR} and α_{CR}. The mutual information $\mathrm{I}(X;Y)$ is then calculated,

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

which is normalised by dividing by $\mathrm{H}(X)$. The resulting averages and standard deviations are displayed in Figure 6. On the whole, it can be seen that the larger a is compared to c, the more utility CR provides compared to GRR. However, this does not tell the whole story, as the difference between datasets has more impact on the utility than the difference between methods.

E. GRR and CR parameter α

To investigate what property of the probability distribution p_{XS} causes CR to outperform GRR, the parameters α_{CR} and α_{GRR} that govern the privacy protocols CR and GRR are considered. Both of these have the property that the higher their value, the less 'random' the protocols are, resulting in a better utility. Since these α are found from ε through different equations, the difference in utility of GRR and CR for different probability distributions may be explained by a difference in α. This assertion is tested for 100 randomly generated distributions in Figure 7. As can be seen, the difference in mutual information can for a large part be explained by a difference in α ($\rho = 0.9815$, $\rho = 0.9889$, and $\rho = 0.9731$, respectively). In Figure 8, the relation between α and the LIP value ε for the experiments in 5(b) and 5(d) is shown. The fact that $\alpha_{\mathrm{GRR}} > \alpha_{\mathrm{CR}}$ in 8(a) corresponds to the fact that GRR outperforms CR in 5(b), and the opposite relation holds between 8(b) and 5(d).

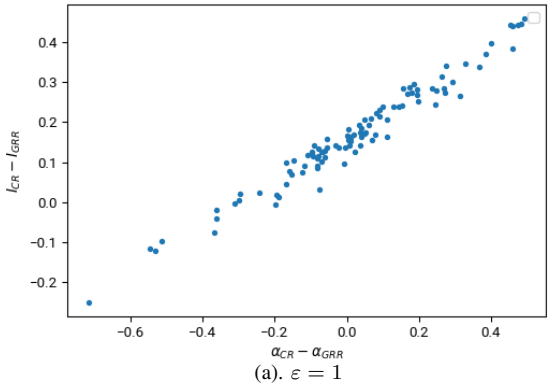
Unfortunately, we were not able to relate the difference in parameter α to other properties of the distribution. Without presenting details we mention that the properties $\mathrm{I}(X;S), \max_{x,s} \mathrm{p}_{x,s}, \max_x \mathrm{p}_x$ and $\max_s \mathrm{p}_s$ do not appear to have an impact on the difference in utility between GRR and CR.

IX. Conclusions and Future Work

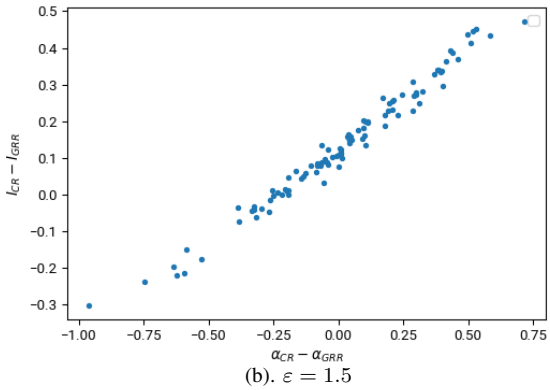
Local data sanitisation protocols have the advantage of being scalable for large numbers of users. Furthermore, the advantage of using differential privacy-like privacy metrics is that they provide worst-case guarantees, ensuring that the privacy of every user is sufficiently protected. For both ε-LDP and ε-LIP methods are derived to find sanitisation protocols that maximise mutual information between input and output, solving the PF problem for these metrics.

Within this setting, it can be observed that ε-LIP has two main advantages over ε-LDP. First, it fits better within the PF setting, where the distribution $\mathrm{p}_{S,X}$ is (at least approximately) known to the estimator. Second, finding the optimal protocol is significantly faster than under LDP, especially for small ε. If one nevertheless prefers ε-LDP as a privacy metric, then it is still worthwile to find the optimal $\frac{\varepsilon}{2}$-LIP protocol, as this can be found significantly faster, at a low utility penalty.

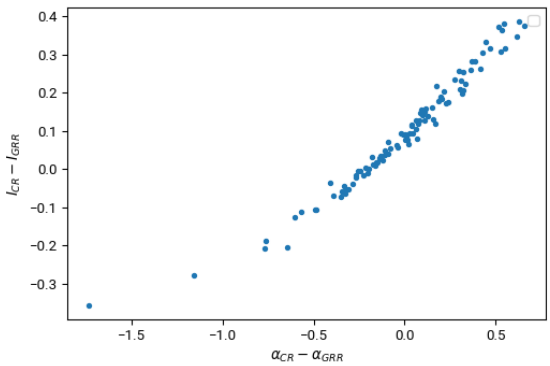
In the multiple attributes setting, it is shown that ε-SRLIP provides additional privacy guarantees compared to ε-LIP, since without this requirement a protocol can lose all its privacy protection in the presence of side channels. Unfortunately, however, experiments show that this is paid for both in computation time and in utility.



(a). $\varepsilon = 1$



(b). $\varepsilon = 1.5$



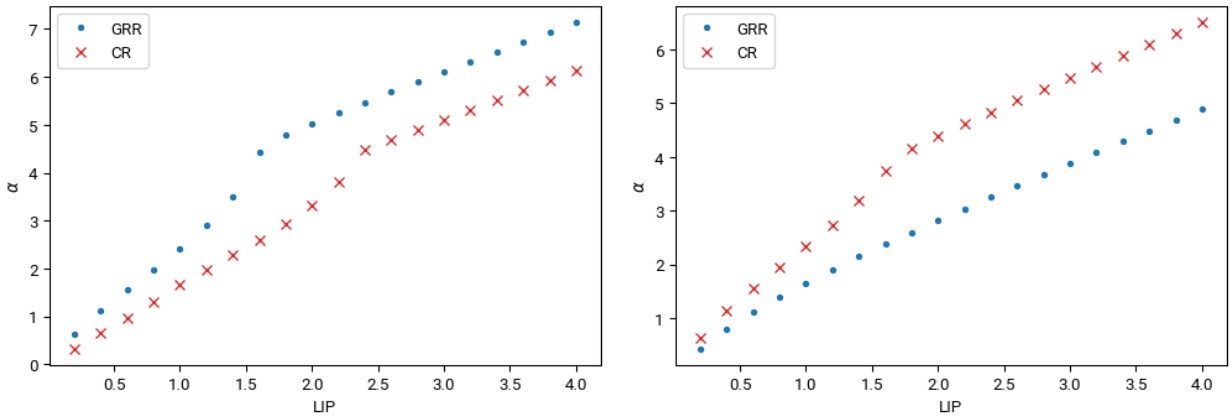
(c). $\varepsilon = 2$

Figure 7. Difference in α versus difference in utility for 100 randomly generated probability distributions, for $a = c = 5$.

With regard to the specific protocols, it is found that the newly introduced protocol, CR, generally outperforms OUE, especially for high values of ε-LIP. This can be explained from the fact that by design the utility of OUE is capped at $\frac{1}{2}\mathrm{H}(X)$. CR behaves more or less similar to GRR, and which of these two protocols performs best depends on properties of the joint distribution $\mathrm{p}_{X,S}$. In particular, it largely depends on which of the two protocols has the highest value of their governing parameter α. Also, it can be seen that CR performs better on average if a is large compared to c.

For further research, a number of important avenues remain

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(a) S = occupation, X = education (b) S = occupation, X = relationship

Figure 8. Value of GRR and CR parameter α for different values of ε for the adult-dataset.

to be explored. First, the aggregator's knowledge about $\mathrm{p}_{S,X}$ may not be perfect, because they may learn about $\mathrm{p}_{S,X}$ through observing $(\vec{S}, \vec{X})$. Incorporating this uncertainty leads to robust optimisation [28], which would give stronger privacy guarantees.

Second, it might be possible to improve the method of obtaining ε-SRLIP protocols via Theorem 4. Examining its proof shows that lower values of ε^j may suffice to still ensure ε-SRLIP. Furthermore, the optimal choice of $(\varepsilon^j)_{j\leq m}$ such that $\sum_j \varepsilon^j = \varepsilon$ might not be $\varepsilon^j = \frac{\varepsilon}{m}$. However, it is computationally prohibitive to perform the vertex enumeration for many different choices of $(\varepsilon^j)_{j\leq m}$, and as such a new theoretical approach is needed to determine the optimal $(\varepsilon^j)_{j\leq m}$ from ε and $\mathrm{p}_{S,X}$.

Third, it would be interesting to see if there are other ways to close the gap between the theoretically optimal protocol, which may be hard to compute in practice, and general LDP protocols, which do not see the difference between sensitive and non-sensitive information. This is relevant because CR needs both S and X as input, and there may be situations where access to S is not available.

Finally, although CR outperforms GRR and OUE for some datasets, it does not do so consistently. More research into the properties of distributions where CR fails to provide a significant advantage might lead to improved privacy protocols.

Acknowledgements

This work was supported by NWO grant 628.001.026 (Dutch Research Council, the Hague, the Netherlands).

References

[1] Milan Lopuhaä-Zwakenberg. "The Privacy Funnel from the Viewpoint of Local Differential Privacy". In: *Fourteenth International Conference on the Digital Society (ICDS)* (2020), pp. 19–24.

[2] Flavio du Pin Calmon, Ali Makhdoumi, Muriel Médard, Mayank Varia, Mark Christiansen, and Ken R Duffy. "Principal inertia components and applications". In: *IEEE Transactions on Information Theory* 63.8 (2017), pp. 5011–5038.

[3] Ni Ding and Parastoo Sadeghi. "A Submodularity-based Agglomerative Clustering Algorithm for the Privacy Funnel". In: *arXiv:1901.06629* (2019). Preprint, accessed 2020.11.8.

[4] Fabian Prasser, Florian Kohlmayer, Ronald Lautenschlaeger, and Klaus A. Kuhn. "Arx-a comprehensive tool for anonymizing biomedical data". In: *AMIA Annual Symposium Proceedings*. Vol. 2014. American Medical Informatics Association. 2014, p. 984.

[5] Ali Makhdoumi, Salman Salamatian, Nadia Fawaz, and Muriel Médard. "From the information bottleneck to the privacy funnel". In: *2014 IEEE Information Theory Workshop (ITW 2014)*. IEEE. 2014, pp. 501–505.

[6] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. "What can we learn privately?" In: *SIAM Journal on Computing* 40.3 (2011), pp. 793–826.

[7] Bo Jiang, Ming Li, and Ravi Tandon. "Local Information Privacy with Bounded Prior". In: *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE. 2019, pp. 1–7.

[8] Salman Salamatian, Flavio du Pin Calmon, Nadia Fawaz, Ali Makhdoumi, and Muriel Médard. "Privacy-Utility Tradeoff and Privacy Funnel". In: *http://www.mit.edu/~salmansa/files/privacy_TIFS.pdf* (2020). Preprint, accessed 2020.11.8.

[9] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. "Extremal mechanisms for local differential privacy". In: *Advances in neural information processing systems*. 2014, pp. 2879–2887.

[10] Borzoo Rassouli and Deniz Gunduz. "On perfect privacy". In: *2018 IEEE International Symposium on Information Theory (ISIT)*. IEEE. 2018, pp. 2551–2555.

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org

[11] Imre Bárány and Attila Pór. "On 0-1 polytopes with many facets". In: *Advances in Mathematics* 161.2 (2001), pp. 209–228.

[12] Naftali Tishby, Fernando C Pereira, and William Bialek. "The information bottleneck method". In: *arXiv:physics/0004057* (2000). Preprint.

[13] SY Kung. "A compressive privacy approach to generalized information bottleneck and privacy funnel problems". In: *Journal of the Franklin Institute* 355.4 (2018), pp. 1846–1872.

[14] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating noise to sensitivity in private data analysis". In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284.

[15] Paul Cuff and Lanqing Yu. "Differential privacy as a mutual information constraint". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016, pp. 43–54.

[16] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. "Our data, ourselves: Privacy via distributed noise generation". In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2006, pp. 486–503.

[17] Cynthia Dwork and Guy N Rothblum. "Concentrated differential privacy". In: *arXiv:1603.01887* (2016). Preprint, accessed 2020.11.8.

[18] Ilya Mironov. "Rényi differential privacy". In: *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE. 2017, pp. 263–275.

[19] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Shuang Song, Kunal Talwar, and Abhradeep Thakurta. "Encode, shuffle, analyze privacy revisited: formalizations and empirical evaluation". In: *arXiv:2001.03618* (2020). Preprint, accessed 2020.11.8.

[20] Daniel Kifer and Ashwin Machanavajjhala. "Pufferfish: A framework for mathematical privacy definitions". In: *ACM Transactions on Database Systems (TODS)* 39.1 (2014), pp. 1–36.

[21] Naoise Holohan, Douglas J Leith, and Oliver Mason. "Extreme points of the local differential privacy polytope". In: *Linear Algebra and its Applications* 534 (2017), pp. 78–96.

[22] David Avis and Komei Fukuda. "A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra". In: *Discrete & Computational Geometry* 8.3 (1992), pp. 295–313.

[23] Milan Lopuhaä-Zwakenberg, Boris Škorić, and Ninghui Li. "Information-theoretic metrics for Local Differential Privacy protocols". In: *arXiv:1910.07826* (2019). Preprint, accessed 2020.11.8.

[24] Mengmeng Yang, Lingjuan Lyu, Jun Zhao, Tianqing Zhu, and Kwok-Yan Lam. "Local Differential Privacy and Its Applications: A Comprehensive Survey". In: *arXiv:2008.03686* (2020). Preprint, 2020.11.8.

[25] Stanley L Warner. "Randomized response: A survey technique for eliminating evasive answer bias". In: *Journal of the American Statistical Association* 60.309 (1965), pp. 63–69.

[26] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. "Locally differentially private protocols for frequency estimation". In: *26th {USENIX} Security Symposium ({USENIX} Security 17)*. 2017, pp. 729–745.

[27] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: http://archive.ics.uci.edu/ml/datasets/Adult.

[28] Dimitris Bertsimas, Vishal Gupta, and Nathan Kallus. "Data-driven robust optimization". In: *Mathematical Programming* 167.2 (2018), pp. 235–292.

APPENDIX A
PROOFS

Proof of Theorem 4. For $J \subseteq \{1,\dots,m\}$ and $j \in \{1,\dots,m\}$, define $J[j] := J \cap \{1,\dots,j-1\}$. Furthermore, write $\mathcal{X}^{\setminus J} = \prod_{j\notin J} \mathcal{X}^j$, and its elements as $x^{\setminus J}$. Define $\varepsilon := \sum_j \varepsilon^j$. Then

$$\mathrm{p}_{y|s,x^J} = \sum_{x^{\setminus J}} \mathrm{p}_{y|x}\, \mathrm{p}_{x^{\setminus J}|s,x^J} \tag{37}$$

$$= \mathrm{p}_{y^J|x^J} \sum_{x^{\setminus j}} \left(\prod_{j\notin J} \mathrm{p}_{y^j|x^j} \right) \mathrm{p}_{x^{\setminus J}|s,x^J} \tag{38}$$

$$= \mathrm{p}_{y^J|x^J} \sum_{x^{\setminus j}} \prod_{j\notin J} \mathrm{p}_{y^j|x^j}\, \mathrm{p}_{x^j|s,x^{J[j]}} \tag{39}$$

$$= \mathrm{p}_{y^J|x^J} \prod_{j\notin J} \sum_{x^j} \mathrm{p}_{y^j|x^j}\, \mathrm{p}_{x^j|s,x^{J[j]}} \tag{40}$$

$$= \mathrm{p}_{y^J|x^J} \prod_{j\notin J} \mathrm{p}_{y^j|s,x^{J[j]}} \tag{41}$$

$$\leq \mathrm{p}_{y^J|x^J} \prod_{j\notin J} \mathrm{e}^{\varepsilon^j}\, \mathrm{p}_{y^j|x^{J[j]}} \tag{42}$$

$$\leq \mathrm{e}^{\varepsilon}\, \mathrm{p}_{y^J|x^J} \prod_{j\notin J} \mathrm{p}_{y^j|x^{J[j]}} \tag{43}$$

$$= \mathrm{e}^{\varepsilon}\, \mathrm{p}_{y|x^J}\,. \tag{44}$$

The fact that $\mathrm{e}^{-\varepsilon}\, \mathrm{p}_{y|x^J} \leq \mathrm{p}_{y|s,x^J}$ is proven analogously. □

Proof of Proposition 1. Write $Q_{y|x,s} = \mathbb{P}(\mathrm{CR}^\alpha(x,s) = y)$. Then

$$Q_{y|x,s} = \sum_{s'} \mathbb{P}(\mathrm{CR}^\alpha(x,s) = y|\tilde{s} = s')\mathbb{P}(\tilde{s} = s'|S = s) \tag{45}$$

$$= \frac{\mathrm{e}^\alpha}{\mathrm{e}^\alpha + c - 1} + \frac{1}{\mathrm{e}^\alpha + c - 1} \sum_{s'\neq s} \mathrm{p}_{y|s'}, \tag{46}$$

where $\delta_{x=y}$ is the Kronecker delta. It follows that

$$\mathbb{P}(\mathrm{CR}^\alpha(X,S) = y|S = s) = \sum_x Q_{y|x,s}\, \mathrm{p}_{x|s} \tag{47}$$

$$= \frac{\mathrm{e}^{\alpha}}{\mathrm{e}^{\alpha}+c-1}\,\mathrm{p}_{y|s} + \frac{1}{\mathrm{e}^{\alpha}+c-1}\sum_{s'\neq s}\mathrm{p}_{y|s'} \tag{48}$$

$$= \frac{\mathrm{e}^{\alpha}-1}{\mathrm{e}^{\alpha}+c-1}\,\mathrm{p}_{y|s} + \frac{1}{\mathrm{e}^{\alpha}+c-1}\sum_{s'}\mathrm{p}_{y|s'}, \tag{49}$$

$$\mathbb{P}(\mathrm{CR}^{\alpha}(X,S)=y)$$

$$= \sum_{s}\mathbb{P}(\mathrm{CR}^{\alpha}(X,S)=y|S=s)\,\mathrm{p}_s \tag{50}$$

$$= \frac{\mathrm{e}^{\alpha}}{\mathrm{e}^{\alpha}+c-1}\,\mathrm{p}_{y} + \frac{1}{\mathrm{e}^{\alpha}+c-1}\sum_{s}\sum_{s'\neq s}\mathrm{p}_{y|s'}\,\mathrm{p}_s \tag{51}$$

$$= \frac{\mathrm{e}^{\alpha}}{\mathrm{e}^{\alpha}+c-1}\,\mathrm{p}_{y} + \frac{1}{\mathrm{e}^{\alpha}+c-1}\sum_{s'}\mathrm{p}_{y|s'}\sum_{s\neq s'}\mathrm{p}_s \tag{52}$$

$$= \frac{\mathrm{e}^{\alpha}}{\mathrm{e}^{\alpha}+c-1}\,\mathrm{p}_{y} + \frac{1}{\mathrm{e}^{\alpha}+c-1}\sum_{s'}(\mathrm{p}_{y|s'}-\mathrm{p}_{y,s'}) \tag{53}$$

$$= \frac{\mathrm{e}^{\alpha}-1}{\mathrm{e}^{\alpha}+c-1}\,\mathrm{p}_{y} + \frac{1}{\mathrm{e}^{\alpha}+c-1}\sum_{s'}\mathrm{p}_{y|s'}\,. \tag{54}$$

It follows that

$$L(\alpha) = \max_{y,s}\left|\ln\frac{\mathbb{P}(\mathrm{CR}^{\alpha}(X,S)=y|S=s)}{\mathbb{P}(\mathrm{CR}^{\alpha}(X,S)=y)}\right|, \tag{55}$$

hence CR^{α} satisfies ε-LIP if and only if $\varepsilon \geq L(\alpha)$. □

2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org