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

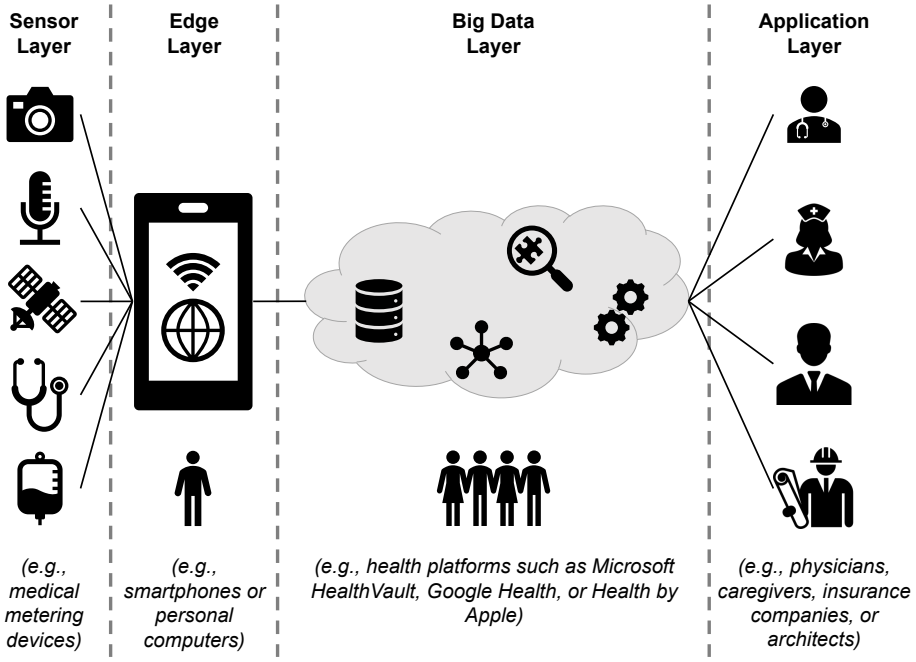


Figure 1. Layered Architecture of an IoT Health Smart Service (cf. [16]).

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

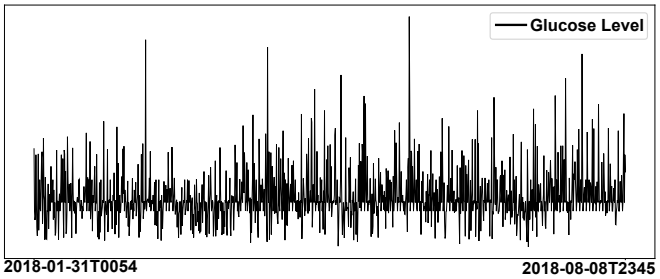


Figure 2. Data from a Continuous Diabetes Monitoring Device.

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

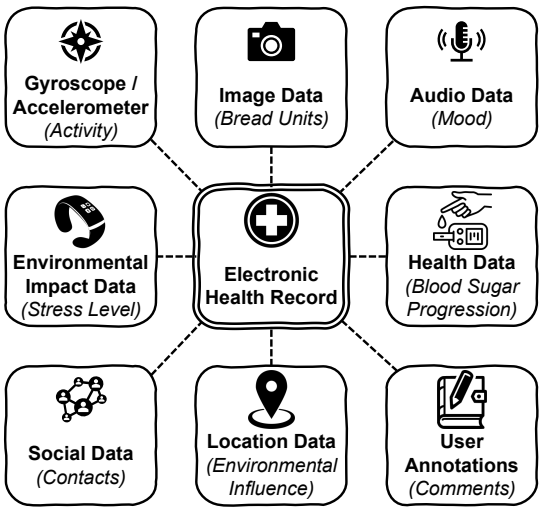


Figure 3. Data Model of an Annotated Electronic Health Record.

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

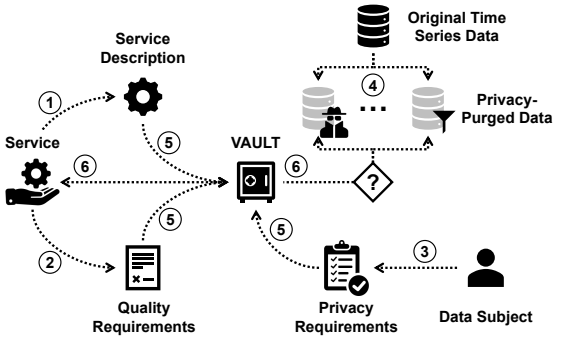


Figure 4. Concept of and Workflow for Data Access via VAULT.

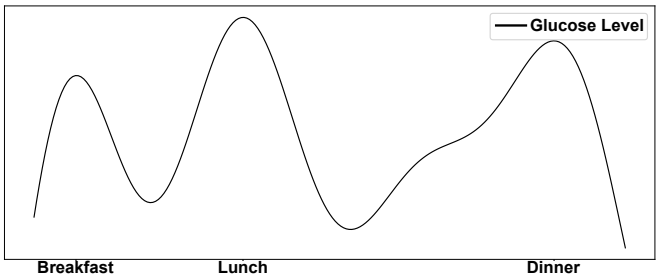


Figure 5. Continuous Diabetes Monitoring Data over the Course of a Day.

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.

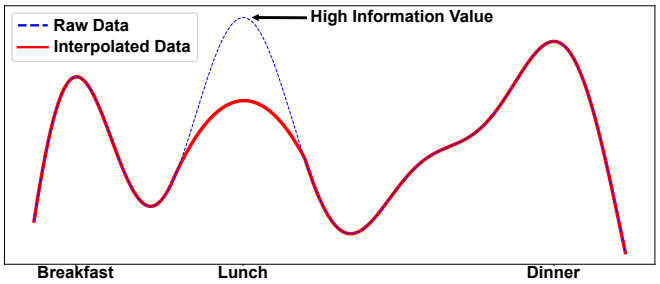


Figure 7. Application of a Spline Interpolation to Time Series Data.

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].

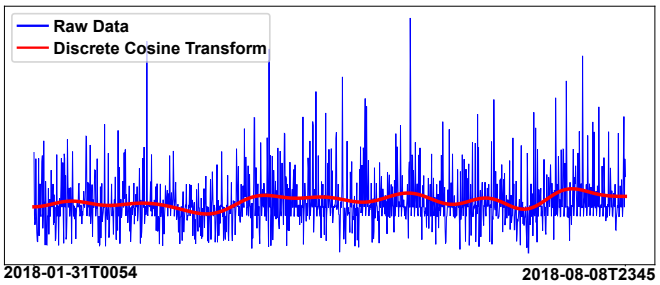


Figure 8. Application of a Fourier Transform to Time Series Data.

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

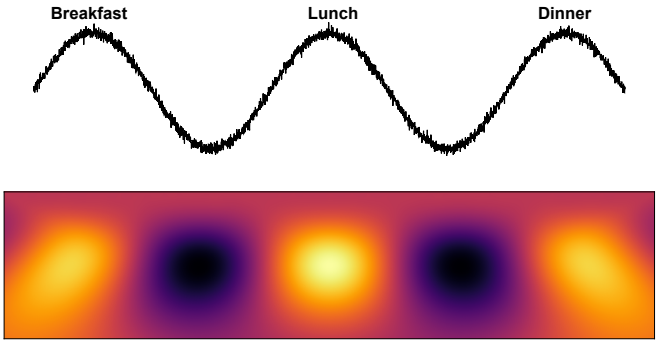


Figure 9. Time-Frequency Representation of Noisy Time Series Data.

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.

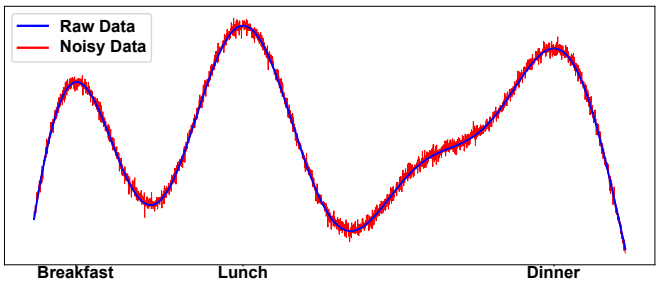


Figure 10. Adding Gaussian Noise to Time Series Data.

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

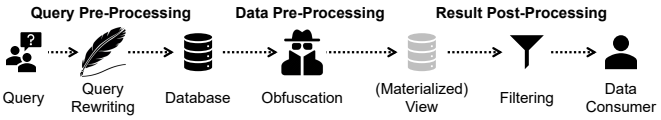


Figure 11. Implementation Strategies for the Privacy Techniques in VAULT.

```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.

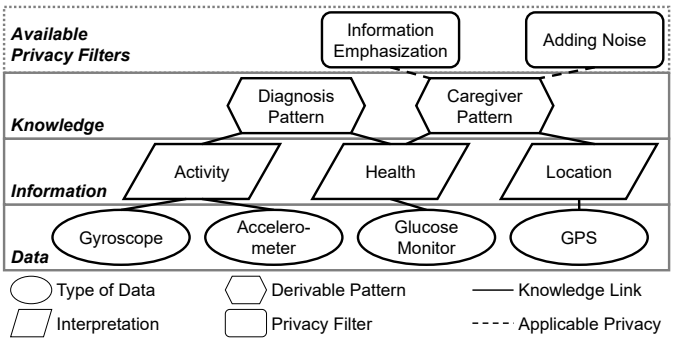


Figure 12. The VAULT Knowledge Model for a Health Record (excerpt).

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

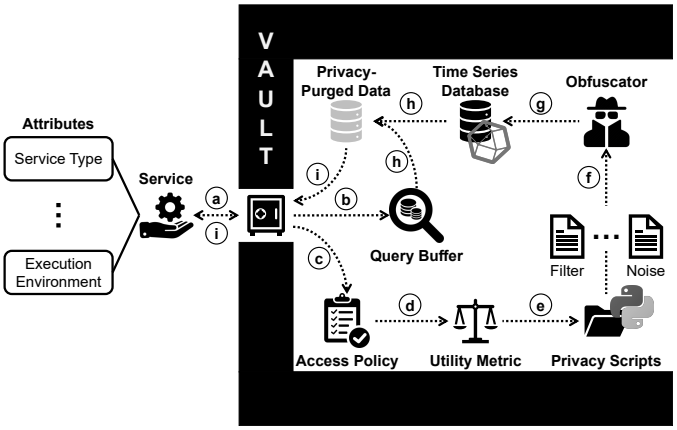


Figure 13. Implementation of and Query Processing in VAULT.

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).

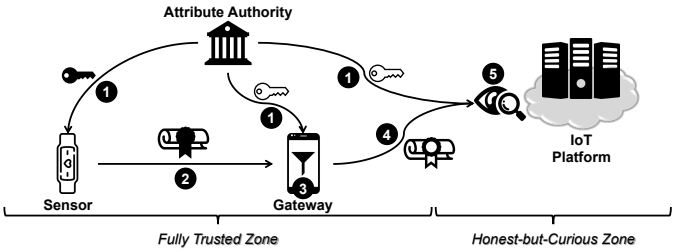


Figure 14. Privacy-Aware Attribute-Based Signature [71].

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).

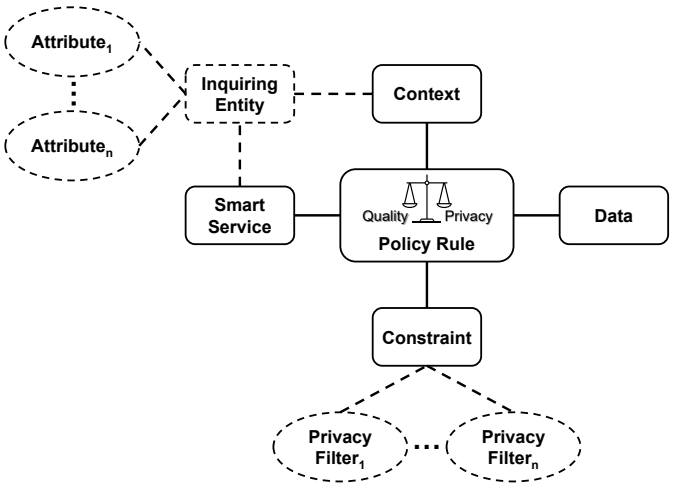


Figure 15. Permission Model Applied in VAULT.

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.

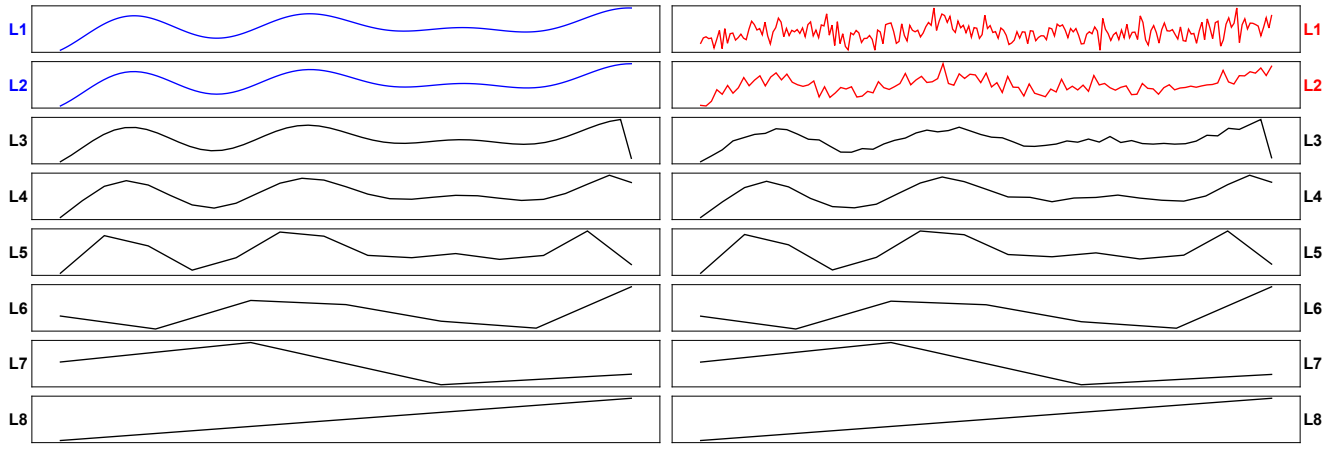
The *Savitzky-Golay filter* is a digital filter which is often used in the field of time series data for smoothing the data

[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].

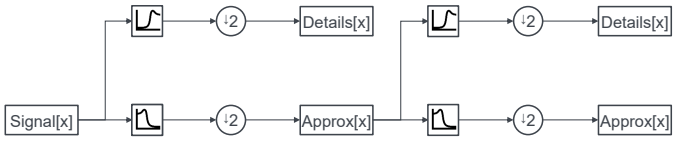
Algorithm 2 shows how the SNIL Algorithm operates. First, the original data series is decomposed into frequency bands using Discrete Wavelet Transform. Then, Gaussian noise is added to all frequency bands between an initial level l_s and an end level l_e. Finally, all frequency bands are merged again 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.

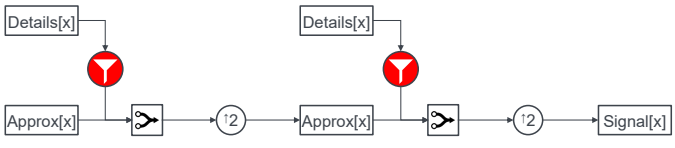
Figure 20 shows the frequency bands of the time series from our running example after applying the SNIL Algorithm. Gaussian noise is added to the frequency bands L2 to L5. The resulting noisy time series is shown in Figure 21. As it can be seen, the utility of the data is preserved—with regard to the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

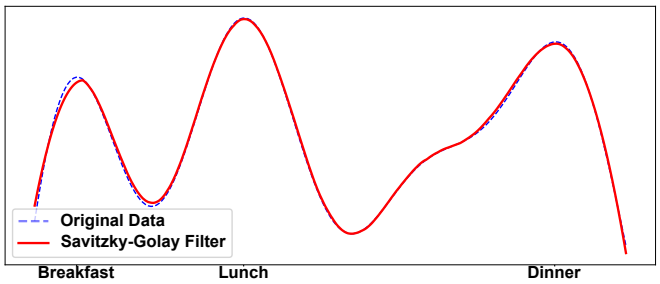


Figure 19. Reconstructed Time Series Using Algorithm 1.

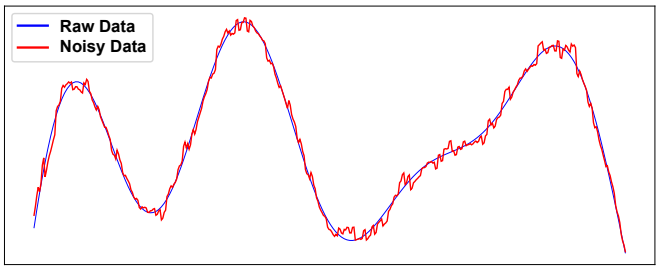


Figure 21. Adding SNIL-Based Noise to Time Series Data.

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;
```

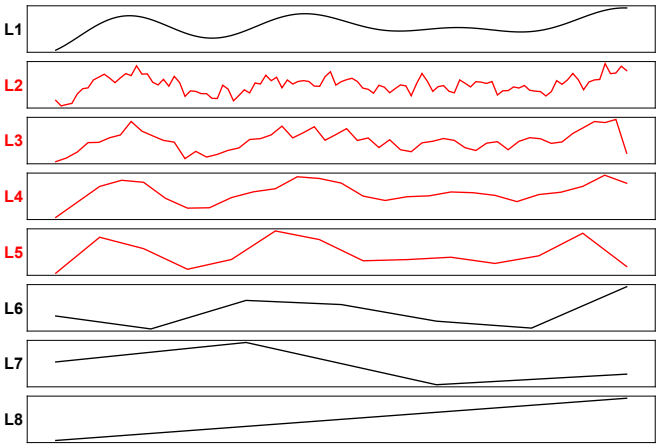


Figure 20. Impact of the SNIL Algorithm on a Time Series.

```
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

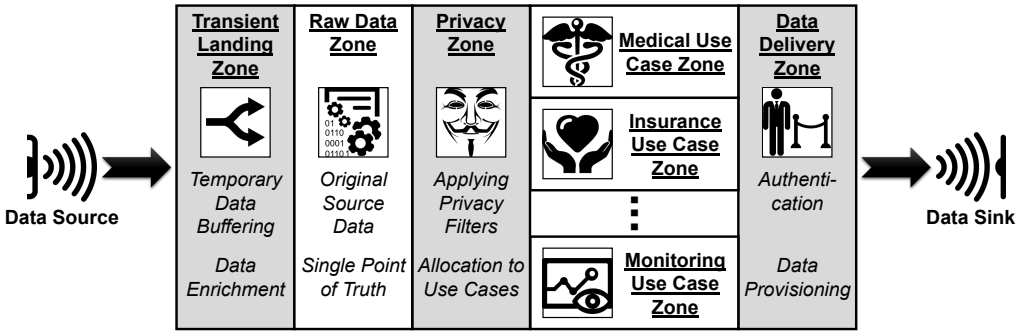


Figure 22. A Multi-Zone Data Lake Architecture for the Data Management in VAULT.

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 $\mathcal{O}(n)$.

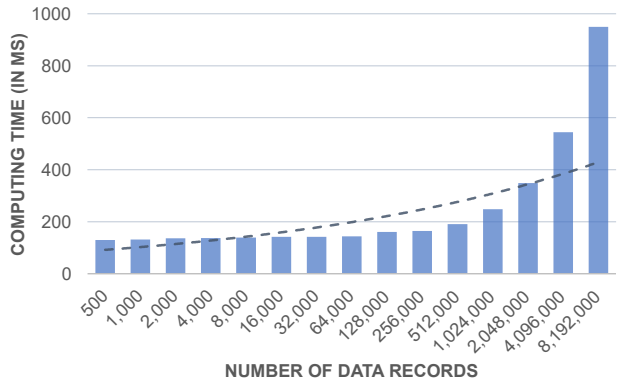
This overhead can also be regarded as runtime overhead for VAULT as a whole (in comparison to a data provision without privacy features) since the privacy filters represent its

[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.

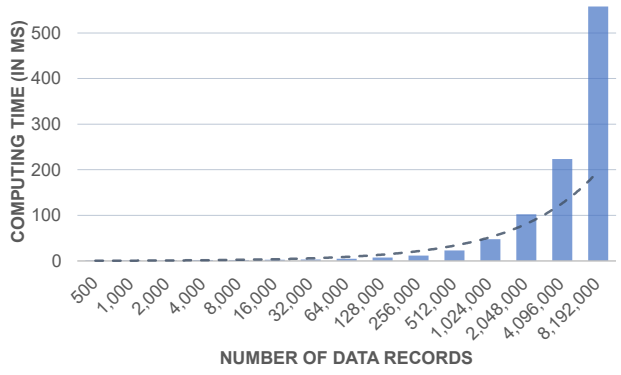
Table III. Performance Analysis Results Overview.

	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

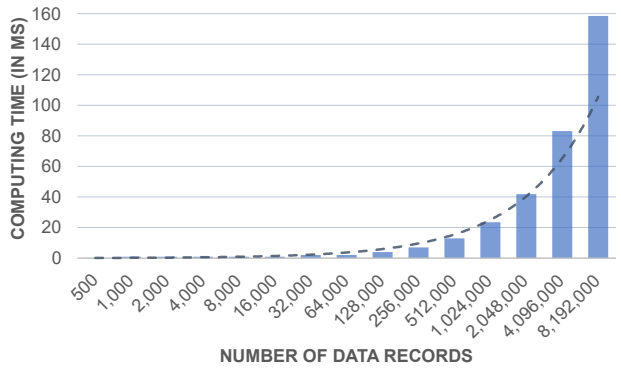
2020, © Copyright by authors, Published under agreement with IARIA - www.iaria.org



(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