PSSST! The Privacy System for Smart Service Platforms
An Enabler for Confidable Smart Environments

Christoph Stach[1], Frank Steimle[1], Clémentine Gritti[2], and Bernhard Mitschang[1]
[1]*Institute for Parallel and Distributed Systems, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany*
[2]*Department of Information Security and Communication Technology, NTNU, Elektro B, Gløshaugen, Trondheim, Norway*
[1]*{forename.lastname}@ipvs.uni-stuttgart.de,* [2]*clementine.gritti@ntnu.no*

Keywords: Privacy, Access Control, Internet of Things, Smart Service Platform, Sensors, Actuators, Stream Processing.

Abstract: The Internet of Things and its applications are becoming increasingly popular. Especially Smart Service Platforms like Alexa are in high demand. Such a platform retrieves data from sensors, processes them in a back-end, and controls actuators in accordance with the results. Thereby, all aspects of our everyday life can be managed. In this paper, we reveal the downsides of this technology by identifying its privacy threats based on a real-world application. Our studies show that current privacy systems do not tackle these issues adequately. Therefore, we introduce *PSSST!*, a user-friendly and comprehensive **p**rivacy **s**ystem for **S**mart **S**ervice Pla**t**forms limiting the amount of disclosed private information while maximizing the quality of service at the same time.

1 INTRODUCTION

The *Internet of Things* (*IoT*) is nowadays monitoring and controlling many aspects of our daily lives. This becomes particularly apparent in the increasingly popular *Intelligent Personal Assistants* such as *Alexa*. These are technical tools that determine user requests via sensors and subsequently fulfill them as effectively as possible via actuators. In the case of Alexa, this is represented by a speaker with a built-in microphone, the so-called *Echo*[1]. Via the microphone, the user can make requests in natural language to Alexa, e. g., "Alexa, what is the weather forecast tomorrow?"). The captured voice message is then sent to the Alexa Cloud service to analyze and interpret it. After the meaning of the question is identified, a corresponding answer is retrieved from a knowledge base. The data are then sent back to Echo and read to the user (Chung et al., 2017b).

As a result of this special kind of interaction, Alexa is no longer seen as a technological gadget, but as a strong social presence (Purington et al., 2017). Therefore, users interact much more naturally with Alexa so that the collected data provide deep insights into their everyday lives (Orr and Sanchez, 2018). Users are therefore increasingly concerned about whether the provider of such a *Smart Service Platform* handles these highly private data in a trustworthy manner. It is also unknown whether only such data are collected which are necessary to provide the specific services. The user cannot verify which data are captured and what knowledge can be derived from them. So, s/he has to rely solely on the good nature of the platform provider. Yet, studies show many potential privacy vulnerabilities in these platforms (Chung et al., 2017a).

On top of this, Alexa shares its data with various third-party Cloud services (e. g., ridesharing providers) and operates several compatible third-party IoT devices (e. g., *Smart Lighting Kits*) (Chung et al., 2017b). That is, even if the provider of the respective Smart Service Platform is entirely trustworthy, its control over the private data ends when such data have been disclosed to these third parties (Alhadlaq et al., 2017).

Samsung even predicts that by 2020 all Samsung household appliances will be interconnected. Then sensors and actuators should be integrated into every device so that these devices can interact with each other autonomously (Samsung Newsroom, 2018).

Although Smart Services provide great convenience for users, the privacy threats they pose are immense (Apthorpe et al., 2016). So, there is an urgent need for privacy systems for Smart Service Platforms. In this respect, it is important to involve the user in all decisions (Rashidi and Cook, 2009). Nevertheless, the autonomous nature of the Smart Service Platform

[1]see https://www.amazon.com/echo

must not be restricted unnecessarily in order to sustain its service quality (Townsend et al., 2011). Due to this balancing act, today's privacy systems fail in the context of Smart Service Platforms: they are unnecessary restrictive (Schaub et al., 2017), are not tailored to the privacy requirements of the users (Felt et al., 2012a), and they overburden the users (Felt et al., 2012b).

For this reason, we introduce *PSSST!*, a novel **p**rivacy **s**ystem for **S**mart **S**ervice Pla**t**forms that is designed to meet the specific requirements of such an ecosystem. To this end we make the following five contributions: **(1)** We provide insight into a Smart Service Platform, namely *MIALinx* (Wieland et al., 2016), and describe how it operates based on an exemplary application scenario. From this, we derive both privacy issues and requirements towards a privacy system in such a context. **(2)** We introduce a mechanism to prevent Smart Devices and Smart Service Platforms from deriving specific knowledge about its users. **(3)** We describe a method to prevent a Smart Service Platform from bypassing this mechanism. **(4)** We ensure a high quality of service despite the privacy measures. **(5)** We combine these three security measures in PSSST!.

The remainder of the paper is structured as follows: In Section 2, an application scenario for a Smart Service Platform is introduced. This exemplary scenario illustrates the functionality of such a platform and its inherent privacy threats. Subsequently, in Section 3, requirements towards a privacy system for such a platform are derived from this scenario. Section 4 discusses existing privacy approaches for Smart Devices and Smart Services. As these approaches do not meet all the requirements towards a privacy system for Smart Service Platforms, Section 5 first introduces the concept of our approach PSSST! before Section 6 describes its implementation. Section 7 assesses PSSST! and reveals whether it fulfills all requirements. Finally, Section 8 concludes this paper.

2 APPLICATION SCENARIO

In the following we describe an example for a realistic *Smart Home* application and identify the components applied in this scenario (see Section 2.1). In this example, an IoT platform, namely MIALinx, is used to provide support for Smart Services. MIALinx evaluates simple *IF-THIS-THEN-THAT* (*ITTT*) rules to react to specific events. Although IoT platforms such as Alexa have a lot of further functionalities, the basic principles of such a platform can be illustrated using MIALinx as an example (see Section 2.2). Based on this scenario, we describe inherent privacy risks of such an IoT platform (see Section 2.3).

2.1 Application Example

Home heating is one of the major energy consumers in a household. Therefore, increasing the efficiency of home heating is an essential goal. With the help of IoT technologies, a need-based heating control is achievable. Figure 1 shows a setup for such an IoT supported heating system. This application example is inspired by the study of Kleiminger et al. (Kleiminger et al., 2014). It reflects the state of the art of technology in the field of *Smart Heating* control.

A Smart Heating is basically a common heating system that can be controlled remotely. For this purpose, it can be accessed e. g. via Bluetooth. In our application example it therefore represents an actuator. Various sensors enable to determine the condition in a Smart Home. For instance, *Smart Thermostats* can be used to determine the temperature in each room. These data are forwarded to a local *Control Unit*. This control unit can thus detect when the temperature in a room drops below a specified threshold. It can then trigger the Smart Heating to reestablish the room temperature as specified by the user.

However, the potential offered by such a *Smart Environment* is considerably farther. Via *Smart Room Management* sensors, such as microphones or cameras, it is possible to determine in which rooms how many people are currently present. In this way, unused or very crowded rooms can be heated less, which significantly reduces energy consumption. However, the coordination and interpretation of all these data sources exceed the capabilities of a user.

The automatic analysis and consolidation of all these sensor data are therefore carried out by an external service provider for the user. The user only specifies certain parameters (e. g., thresholds for his or her individual comfortable temperature) and the service provider generates corresponding rules. In addition, the service providers have the capacities to store the history data of their users and to perform analyses on it. This enables them to learn more about the habits of their users (e. g., the user is never at home on the weekend) and to tailor their services accordingly (e. g., the room temperature can be decreased at weekends).

External service providers have another decisive advantage: they are able to join data from different Smart Environments. In our application example, we chose the user's *Smart Car*. Modern Smart Cars are able to identify the driver, e. g., via individual keys for each driver. In addition, such cars are not only able to determine their current location, but they have also access to various data sources. For instance, they can retrieve real-time traffic information, have access to navigation data, and know the appointments a user has

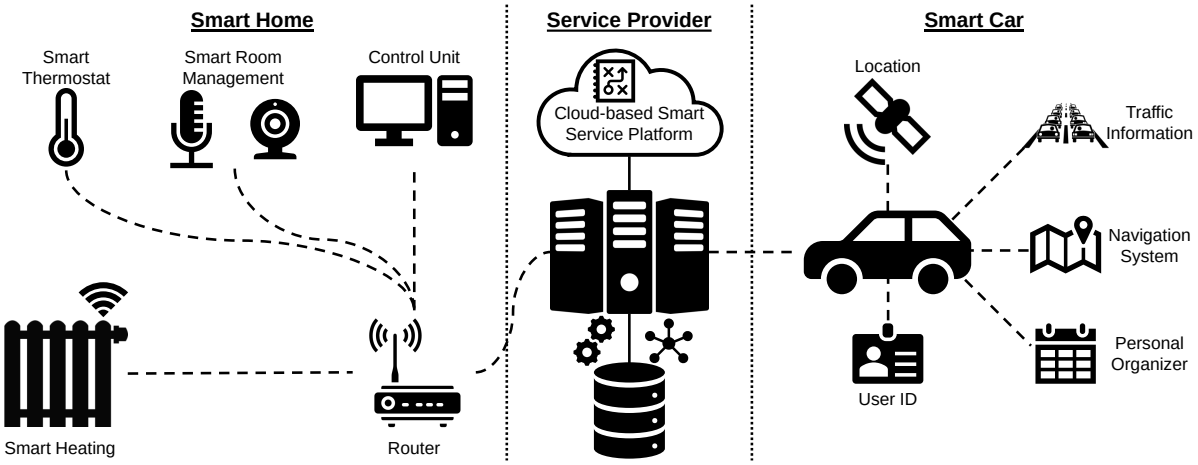


Figure 1: Setup of an IoT Supported Smart Heating System.

stored in his or her personal organizer. This enables them to know when a user is driving home and when s/he is likely to arrive there. By sharing this information with the service provider, the Smart Heating can be adjusted so that the temperature requested by the user is reached as soon as s/he enters his or her home.

This application example illustrates that Smart Services have to process data from various sensors and have to be able to react to certain sensor values. A variety of actuators must be triggered for this purpose. Furthermore, it is evident that a platform for such services must interact with and coordinate multiple heterogeneous Smart Environments. A very simple example of how such a platform operates is described below based on MIALinx. Yet, the insights also apply to other Smart Service Platforms, such as the Alexa platform mentioned in the introduction.

2.2 MIALinx

MIALinx (Wieland et al., 2016) is a Smart Service Platform designed for Industry 4.0. It provides a flexible and easy-to-use integration of assets on the shop floor. The platform allows the creation and execution of ITTT rules. Such a rule describes how to react to certain sensor values, i. e., which actuators have to be addressed. Despite its focus on an industrial context, MIALinx can be applied to any given scenario.

The operating principle of MIALinx is shown in Figure 2. MIALinx introduces an adapter concept in order to be compatible to any given sensor and actuator. An adapter is a *digital twin* for actual physical assets. This means that for each sensor and actuator MIALinx requires such an adapter, which can be seen as a driver which can be added to the platform at runtime. The adapters read the data from their corresponding sensors and convert these raw data into values that can be processed by MIALinx. For instance, the adapter of a Smart Thermostat can convert its raw data into degrees Celsius. Likewise, the actuator adapters receive commands from MIALinx and convert them into instructions that are executable by the physical actuators. A single sensor can feed several adapters and a single adapter can combine the data of several sensors (this also applies to the actuators). For instance, an adapter can represent a virtual Smart Room Management sensor analyzing both camera data and microphone data and determines how many people are in the room by combining both types of data. Then, the adapter passes only the amount of people to core of MIALinx, its rule engine. Here virtual sensors are linked to virtual actuators.

This linkage is realized via ITTT rules, where the left side (IF) represents a virtual sensor and the right side (THEN) a virtual actuator. In addition, a condition is attached to the sensor values under which the rule has to be executed. In the example given in Figure 2, the heating should be turned down when the temperature is above 25° C and it should be increased when the user arrives at home in less than 15 minutes. Machine learning techniques can be applied to this rule base to come up with new rules or to refine existing ones.

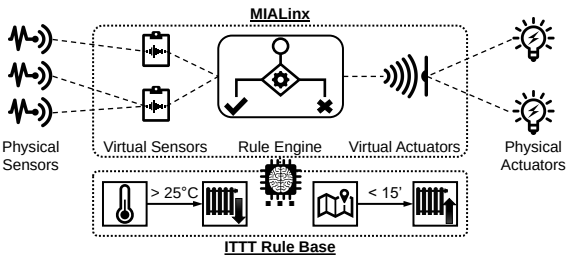


Figure 2: Basic Concept of a Smart Service Platform Exemplified by MIALinx (cf. (Stach et al., 2018c)).

2.3 Privacy Threats

This application example reveals various privacy threats due to the use of a Smart Service Platform:

PT_1: The sensors share their data unrestrictedly with the external service provider. The user cannot track or control which data are transmitted (i. e., which data are leaving his or her sphere of influence). For instance, s/he cannot control whether the Smart Room Management component only passes on the number of people in a room or also its video and audio recordings.

PT_2: The service provider may add further rules to the rule base at any time (e. g., via machine learning). While this is necessary to improve the service quality, it can also be misused to spy on the user. For instance, a rule could be added which sends an email to the service provider, informing him or her when and how long the user leaves the house.

PT_3: The service provider is also able to control all actuators at will. Depending on the actuator, this can cause high costs or even hazards. For instance, the user cannot restrict access to the heating system (e. g., set a limit for operating hours at the heating system).

PT_4: The communication between the sensors and actuators is not secured and can be intercepted.

PT_5: Any data source can forward data to the service provider. An attacker could therefore pretend to be the user's thermostat and use fake data to cause an incorrect control of the heating system.

PT_6: Sensors for home use are produced very cheap and therefore often provide temporary wrong values. This can trigger the wrong rules if the service provider does not filter out such outliers.

3 REQUIREMENTS

These privacy threats result directly in requirements towards a privacy system for Smart Service Platforms:

R_1: To prevent **PT_1**, a privacy system has to be able to control the use of private data as close to sensor as possible. The restriction method must be adapted to the respective sensor, e. g., reducing the accuracy makes sense for location data whereas this method cannot be used for the user ID. Overall, the best privacy strategy is not to pass on any data which are not required for the intended Smart Service to the processing platform.

R_2: To prevent **PT_2**, a privacy system has to be able to monitor and control all data processed by a Smart Service Platform. Only at the platform, all data are available which are processable by the Smart Services. All data sources have to be known to the privacy system to enable it to analyze which knowledge can be derived from the data. In addition to the real-time sensor data, also the knowledge base of the platform (i. e., history data) has to be taken into account.

R_3: To prevent **PT_3**, similar to **R_1**, also the use of actuators has to be restricted locally at the actuator. That is, the interface of actuators must be manipulated, e. g., disable a function or restrict the use of a function.

R_4: To prevent **PT_4**, the communication channels between the sensors / actuators and the Smart Service Platform must be fully encrypted.

R_5: To prevent **PT_5**, all sensors / actuators have to authenticate to the Smart Service Platform.

R_6: To prevent **PT_6**, all sensor data have to be validated sensor-sided, i. e., before passing it on to the Smart Service Platform. The options to deal with outliers largely depend on the respective sensor.

In addition, there are further requirements towards such a privacy system in order to be effective:

R_7: The quality of the Smart Services must be retained. If no data are shared with the platform, privacy is fully protected, but the quality of service is nonexistent. Whereas if all data are shared with the platform, the quality of service is maximized, but there is no privacy. Therefore, an adequate trade-off has to be achieved.

R_8: A Smart Service Platform is a dynamic execution environment—new sensors and actuators can be added at any time. So, a privacy system has to be extendable to cope with such structural and technological changes.

R_9: The configuration of the privacy system has to be simple, as most users of Smart Services are no IT experts. For this purpose, the specification of privacy requirements must not be at the sensor level, but at a higher level. That is, the user describes which knowledge about him or her has to be concealed (e. g., the platform must not detect that the user is never at home on weekends)—s/he does not want to worry about which sensors might reveal this information.

4 RELATED WORK

The privacy approaches which are applicable to Smart Service Platforms can be divided into four categories: attribute-based privacy rules, context-sensitive privacy rules, privacy via mock data, and statistical privacy techniques. Some approaches have features of several categories. When discussing related work, we do not differentiate between approaches for Smart Devices and approaches for IoT back-ends, as both have to be considered by a privacy system (see **$R_1 - R_3$**). The overview is not intended to be exhaustive due to the sheer number of similar approaches. We list representatives for the respective categories, only.

Attribute-based Privacy Rules. Attribute-based privacy control is a relatively simple approach. In this privacy technique, the user defines which attributes a third party gets access to. In the simplest case, one attribute comprises all data of a certain sensor class. Transferred to the application example in Section 2.1, a user could prohibit that GPS data are forwarded to the Smart Service Platform. *DEFCON* (Migliavacca et al., 2010) is such a privacy system for data streams. All events of a restricted sensor are not processed or forwarded—i. e., it appears that these events never occurred. That can also be applied to databases. Kabra et al. (Kabra et al., 2006) introduce a fine-grained access control for database queries. Specific rows or columns can be excluded when processing a query, i. e., particular readings as well as sensor attributes (e. g., latitude or longitude) can be concealed. *Dr. Android and Mr. Hide* (Jeon et al., 2012) proceeds more fine-grained, as it also takes the data format of the respective source into account. Depending on the data format, the user has differently detailed restriction options.

Overall, all of these approaches have a common shortcoming. Their attribute-based data filtering is highly restrictive. That is, a user either shares his or her GPS data with a third party or s/he does not. However, with this technique it is not possible to disclose only when a user will arrive at home without sharing his or her location permanently. So, either the privacy protection or the service quality is heavily impaired.

Context-sensitive Privacy Rules. A less restrictive access control mechanism is enabled by context-based privacy rules. This means, each access rule is linked to an activation context. So, it is possible to share a user's GPS data only when s/he is close to his or her house, i. e., only when it is relevant for controlling the Smart Heating. *ACStream* (Cao et al., 2009) implements such an access control for data stream systems. While in ACStream the context refers only to the content of the transmitted data, *CRePE* (Conti et al., 2010) uses any available sensor data to define the context. However, this leads to an increased complexity regarding the description of the context. To reduce this complexity, Wallis et al. (Wallis et al., 2018) take the application domain into account and consider only aspects which are relevant in the respective domain.

However, all these approaches have the problem that they either consider only spatiotemporal data as context or they completely overburden users with the definition of a more complex context.

Privacy via Mock Data. Approaches of the third category do not restrict access to data sources but enable users to reduce the data quality when necessary. In this way it is possible to provide less trustworthy recipients with manipulated or even completely randomized data. *MockDroid* (Beresford et al., 2011) implements this in a very minimalistic manner—the recipient only receives empty result sets. *AppFence* (Hornyack et al., 2011) therefore adopts a more comprehensive approach towards data shadowing. Instead of empty result sets, plausible mock data are returned—i. e., the data are within the expected value range. For instance, AppFence provides valid latitude and longitude data when mocking a real GPS source. However, these values still might be implausible (e. g., they might point to a location in the middle of the ocean). *PRIVACY-AVARE* (Alpers et al., 2018) therefore takes the properties of the respective source into account. It introduces a specific mocking technique for each data source. For instance, when the accuracy of location data are reduced, it is ensured that properties such as direction of motion or traveled distance are preserved.

Yet, the impact on service quality is unpredictable if data are arbitrarily manipulated, e. g., if the predicted number of people in a room is incorrect, it is unclear how this affects the Smart Heating. Moreover, the intentional manipulation of data is a breach of contract, if this results in a monetary advantage for the user.

Statistical Privacy Techniques. Statistical Privacy Techniques follow a completely different approach. Instead of focusing on the data of a single user, these approaches consider larger user groups. The basic idea is that in this way an individual person vanishes into the crowd. Thereby statistical information about the dataset as a whole can be obtained without being able to draw conclusions about individual users. Sopaoglu and Abul (Sopaoglu and Abul, 2017) introduce a technique to ensure *k-anonymity* for big data processing platforms—i. e., each record cannot be distinguished from at least $k-1$ other records. Yet, k-anonymity does not guarantee that each group of k records is heterogeneous whereby knowledge about an individual user can still be gained. *DAHOT* (Kotecha and Garg, 2017), a privacy approach for data streams, prevents such homogeneity attacks by ensuring the *l-diversity* property as well—i. e., the intra-group diversity in each group of k records is guaranteed. In order to prevent an attacker from exploiting prior knowledge to identify an individual within a group, *FLEX* (Johnson et al., 2018) applies *differential privacy* (Dwork, 2006).

Yet, this cannot be applied to Smart Service Platforms. These platforms have to be able to identify and address each sensor and actuator individually—the service must not activate k heating systems only because one of the k users is on his or her way home.

An inherent problem of all these approaches, however, is their strict focus on raw data. Users, on the other hand, think at a higher level of abstraction regarding their privacy—they know which information

they want to conceal but not which sensor data might reveal this information. Therefore, users are not able to specify their privacy requirements properly (Barth and de Jong, 2017). Also, from a technical point of view, these approaches are not capable of addressing complex privacy requirements adequately. In such cases, the privacy settings have to be much too restrictive and thus impair the service quality unnecessarily.

5 CONCEPT OF PSSST!

Since on the one hand none of the existing privacy solutions is suitable for Smart Service Platforms (see Section 4) and on the other hand such a platform poses a great threat potential (see Section 2.3), we have developed a concept for a novel privacy system for Smart Service Platforms called PSSST!. Its basic architecture with its three key components is shown in Figure 3.

PSSST! applies a two-layered privacy approach. On the one hand, initial privacy control and access restriction measures are enforced directly at the sensor and actuator level. On the other hand, all communication with the Smart Service platform is secured via a second access control layer. In this way PSSST! brings together the best of both worlds. The privacy components for the sensors and actuators are able to monitor data where they are produced. In this way it is ensured that no sensitive data are accessible from these data sources and no third party is able to access critical features of an actuator. In this layer, however, only a very restrictive privacy control can be applied, whereas the second layer knows the entire execution context (all involved data sources) and is therefore able to carry out a much more fine-grained access control.

This second layer has to be implemented in the *Edge*, i. e., on a central processing node of the respective Smart Environment. As several Smart Environments can provide data for a Smart Service, the infrastructure of the Smart Service Platform provider would be the best execution environment for the privacy measures of this layer, from a technical point of view. From a privacy point of view, however, this is out of the question—this would put the fox in charge of the henhouse. As a consequence, several *Secure Access Control Layers* (one per Smart Environment) might be responsible for a certain Smart Service. There is therefore a Cloud-based master instance for the configuration and coordination of these layers. These three components are detailed in the following.

Privacy Control for Sensors and Actuators. As the MIALinx example showed (see Section 2.2), there is always a digital twin for each physical sensor or actuator, via which a Smart Service Platform can interact

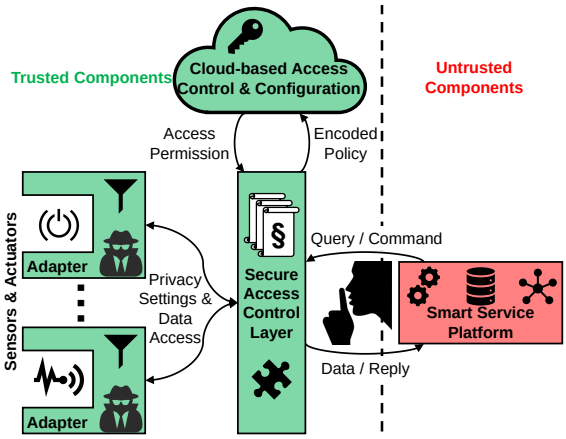


Figure 3: Basic Architecture of PSSST!.

with the respective component. This is where PSSST! steps in. This software has to be installed in the Smart Environment which hosts the corresponding hardware. That is, it is not controlled by Smart Service Platform provider and so it can be enhanced.

These privacy extensions of PSSST! integrate the features of attribute-based privacy rules and privacy via mock data approaches. Thereby, for each sensor the transmission of all data can be blocked—to the outside, it appears as if the sensor is not sending any data. For actuators, the public interface can be restricted, i. e., less functions are available to the public. It is also possible to manipulate individual sensor attributes, e. g., reduce the accuracy of location data. Likewise, it is possible to reduce the functionality of an actuator, e. g., the Smart Heating cannot be set to a temperature above 30° C. Since adapters have to be tailored to the respective sensors or actuators anyway, it is possible in PSSST! to adapt the privacy techniques to the sensors and actuators similar to PRIVACY-AVARE.

The adapters are also able to verify sensor readings. As adapters know the technical specification of their corresponding sensors, they are able to detect outliers. Technical experts specify how to deal with corrupted sensor values in such cases. For instance, these values can be dropped, replaced by the last known valid value, or corrected (e. g., replaced by the average value).

As the study of related work reveals, this approach is unnecessarily restrictive. It is not possible to apply context-based rules as each sensor only knows its own data. Thus, outsourcing data processing entirely to Smart Devices, as practiced in *Secure Function Evaluation* (Mood and Butler, 2018), is not an option. This approach also does not reflect the way people think about privacy as it is based exclusively on data sources. So, the PSSST! adapters must not be configured directly by the users. Rather, they are configured by the central Secure Access Control Layer. This layer not

only holds the context knowledge from all data sources, but also the view on privacy is more user-friendly.

Privacy Control for Smart Environments. We apply a novel pattern-based access control mechanism in the Secure Access Control Layer of PSSST!. The term *pattern* refers to a certain chronological sequence of sensor values. Each of these patterns can be assigned to a certain implication. An example for such a pattern is that the location of a user is in his or her house and then the number of people in the house increases. This pattern could indicate that the user is having a party. As the detection of a pattern is not necessarily a bad thing (it can either be used to spy on the user or to control the Smart Heating), we distinguish between *public patterns*, that should be detectable by a Smart Service Platform, and *private patterns*, that have to be concealed. The following example illustrates the difficulties of this approach. Assuming A, B, and C are sensor readings and the following data are transmitted:

$$A_{t=5} - B_{t=4} - A_{t=3} - B_{t=2} - C_{t=1}$$

The indices specify the timestamps of the readings, e. g., $C_{t=1}$ was transmitted immediately before $B_{t=2}$. Let P_{priv} be a private pattern and P_{pub} a public pattern:

$$P_{priv} = C \rightarrow B \rightarrow A \qquad P_{pub} = A \rightarrow B \rightarrow A$$

P_{priv} could be concealed by dropping $A_{t=3}$. However, this would also prevent the detection of P_{pub} at the same time. P_{priv} could also be concealed by dropping $B_{t=2}$ without affecting the detection of P_{pub}. Yet, if an attacker knows that B occurs at any even timestamp, s/he can still detect P_{priv}. Different techniques for concealing private patterns have to be considered and the one that has the least impact on the recognition of public patterns (i. e., the service quality of the Smart Service Platform) has to be chosen by PSSST!.

As a Smart Service Platform can only access sensors and actuators via the Secure Access Control Layer, these components have to register at that layer first. This registration process involves an authentication step to ensure that no unauthorized components send data to or receive messages from the platform.

However, this approach has the inherent problem that each Smart Environment hosts its own instance of this layer. To allow users to specify their privacy requirements only once, and to ensure that these requirements are subsequently applied to all environments, a central control unit is required to coordinate them.

Cloud-based Access Control and Configuration. Studies by Zheng et al. (Zheng et al., 2018) show that Smart Home users are totally overburdened with the specification of their privacy requirements as they cannot comprehend how their data are processed. On the one hand, they fear that if they apply a particular privacy setting, they will no longer be able to use a certain Smart Service. On the other hand, they are not aware of which data are collected at all.

For this very reason, we adopted a different approach in PSSST!. Users describe their privacy requirements at a high level in natural language. They solely focus on knowledge that must not be disclosed to third parties without considering which data sources are available to the Smart Service Platforms. Technology and domain experts then translate the requirements into private patterns. These experts know in detail what data sources are available in a given Smart Environment and what knowledge can be derived from them. In addition to the privacy requirements, users also specify the service quality they expect from a Smart Service Platform. The technology and domain experts define public patterns from these quality requirements.

The specification of quality requirements is not mandatory for the user. It can also be done through the Smart Service Platform, for example. But even if no public patterns are defined, PSSST! still can be used. The public patterns only help to maximize the service quality of Smart Services (see Section 6.2).

Once all patterns have been specified, they are converted into configurations for the adapters and Secure Access Control Layer and deployed to all Smart Environments. If new sensors or actuators are added to an environment, the central control unit is informed and there the configurations are adjusted accordingly.

6 IMPLEMENTATION OF PSSST!

To realize PSSST!, we rely on established technologies, namely the *PMP* (Stach and Mitschang, 2013), *PATRON* (Stach et al., 2018b), and *CHARIOT* (Gritti et al., 2018). Yet, PSSST! is more than the sum of the positive properties of these three technologies. The concept of PSSST! yields synergy effects that not only improve privacy and service quality, but also significantly reduce the computational overhead.

First, we detail in Section 6.1 how we can integrate the PMP into the digital twins to enforce privacy and at the same time validate the data. In Section 6.2 we address how PATRON realizes the privacy control for Smart Environments. Then, we outline how the policy-based access control mechanism of CHARIOT can be used to authenticate the sensors and actuators in Section 6.3. Finally, in Section 6.4 we describe how PATRON's tool support can be applied to the configuration mechanism of PSSST! in order to reduce the experts' effort and automatize most of their tasks.

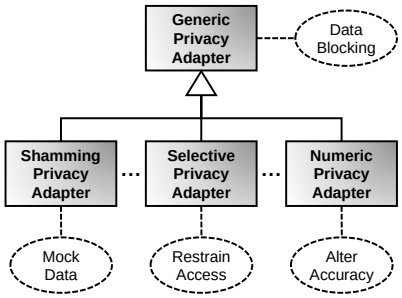
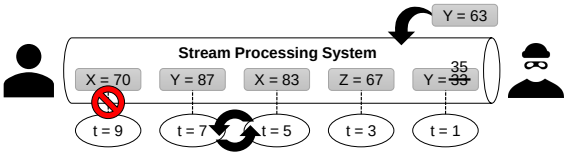
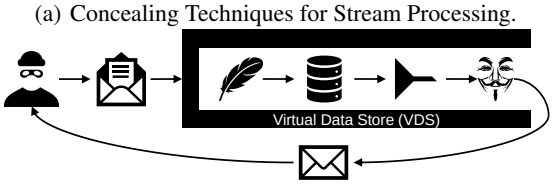


Figure 4: Classification of PSSST! Adapters.



(a) Concealing Techniques for Stream Processing.



(b) Concealing Techniques for Data Stores.

Figure 5: Pattern-based Privacy Control in PSSST!.

6.1 The PMP Resources

The PMP (Stach and Mitschang, 2013) is a privacy mechanism for Smart Devices. In a nutshell, the PMP deploys content providers on the Smart Device that run as background services. These content providers are called *PMP Resources*. Each resource is responsible for a specific type of data (e. g., location data, health data, or contact data). The PMP ensures that applications only have access to the respective data via the Resources. For each data access, the responsible Resource checks whether the requesting application is allowed to query these data. Yet, the PMP not only restricts access to data, but also access to critical system functions. To this end, Resources are used as service providers via which system functions are invoked.

Unlike many similar approaches, the PMP not only enables users to define whether a certain request should be executed or rejected. Rather, data access can be restricted at a fine-grained level. Depending on the protected data source or system functionality, there are different classes of PMP Resources. For instance, there are Resources that support the randomization of data, Resources that block certain data attributes, or Resources that reduce the accuracy for numerical data.

Since the PMP is extendable, i. e., its plugin concept allows to add Resources at runtime, it is feasible to create and apply specialized Resources dedicated for a specific use-case. For instance, Stach (Stach, 2018) introduces a Resource, that enables users to control to which domains an application may send data.

As the adapters of a Smart Service Platform have a similar functionality, we can adopt the access control features of the PMP Resources in PSSST!. That is, the adapters verify before every data access whether it satisfies the privacy policy. Figure 4 shows the privacy-aware adapter classes, that we introduced in PSSST!. Each adapter is able to block access to its data plus there are adapters to provide mock data, to conceal certain attributes, and to reduce the accuracy of the data. Specialized adapters can be deployed if needed.

6.2 The PATRON Execution Layer

PATRON (Stach et al., 2018b) is a privacy system for the IoT. It provides an access control tier isolating data sources from data consumers, i. e., it is able to monitor the entire data transfer. To ensure privacy, PATRON also relies on private patterns that have to be concealed and public patterns that have to be detectable. Various techniques can be used for this purpose (Zhang et al., 2017). PATRON selects the technique that results in the best service quality, i. e., the one causing the least *false positives* (a public pattern is wrongly detected) and *false negatives* (a public pattern is not detected). To this end, the following quality metric is used:

$$QM = \sum_i public\_pattern_i * w_{pub\_i} - \sum_j false\_positive_j * w_{false\_j} - \sum_k private\_pattern_k * w_{priv\_k}$$

Hence, the detected public patterns are counted and the false positive as well as the revealed private patterns are subtracted. Each factor is individually weighted, e. g., to declare a private pattern as less privacy critical. For each configuration (i. e., a selection of concealing techniques), this quality metric is calculated and the one with the highest value is applied in PATRON.

Maximizing this metric is very computation-intensive. As PSSST! filters data in its adapters already, the calculation is accelerated considerably.

The selected configuration is then applied to both real-time and history data (see Figure 5). As shown in Figure 5(a), data can be dropped, the time stamps of data can be swapped, additional data can be injected into the data stream, or data values can be changed to conceal private patterns in data streams (Palanisamy et al., 2018). The concealing procedure for data stores is shown in Figure 5(b). An abstraction layer (*VDS*) is introduced for this purpose. Externally, the VDS

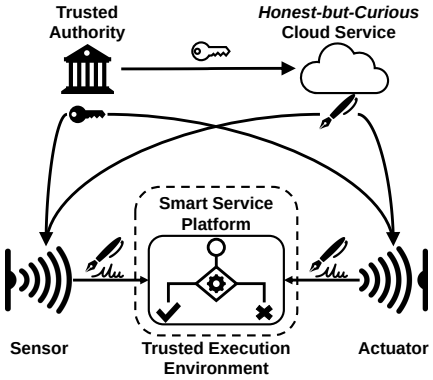


Figure 6: Adapter Authentication (cf. (Gritti et al., 2018)).

acts like the original data store. Internally however, the VDS rewrites incoming queries, adds filters, and applies anonymization functions to the query results to conceal private patterns (Stach and Mitschang, 2018b).

6.3 The CHARIOT Protocol

In the IoT, most of the Smart Devices have very limited resources in terms of memory and computing power. For this reason, many sophisticated security mechanisms cannot be applied to them. CHARIOT (Gritti et al., 2018) addresses this problem. To this end, a server-aided access control mechanism is introduced. Via this mechanism IoT platforms are enabled to determine whether a certain Smart Device has the required credentials to transmit its data to the platform or fetch data from the platform. For this, it is necessary that the devices authenticate to the platform. To ensure that this does not compromise a user's privacy (most Smart Devices are explicitly linked to a user), an *Attribute-Based Signature* is used (Gritti et al., 2019). Since the computation of this signature is computation-intensive, CHARIOT outsources such operations to the Cloud to reduce the Smart Devices' workload.

Figure 6 shows how CHARIOT can be applied to PSSST!. Initially, a trusted authority creates a public / private key pair for each sensor and actuator. An *outsourcing key* is sent to a Cloud service which is considered as *honest-but-curious*, i. e., it is not fully trusted. Whenever a sensor or actuator interacts with the Smart Service Platform (via the Secure Access Control Layer), it needs to sign all messages using its attributes. The recipient can then verify the signature without disclosing the identity of the sender. As this signing process is costly, it is outsourced to the Cloud service. This service operates with the outsourcing key and the sensor or actuator only has to finalize the signature with its private key. Due to this distributed approach, the Cloud service cannot access the data.

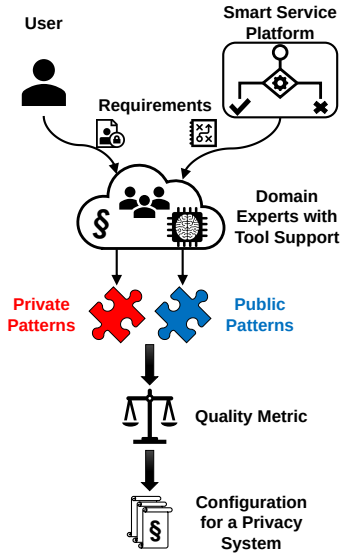


Figure 7: PSSST! Configuration Process.

This solves the problem that PATRON's access control tier cannot verify the identity of sensors and actuators. In addition, by applying CHARIOT in PSSST!, its asynchronous encryption can also be used to secure all communication channels.

6.4 The PATRON Configuration Layer

In PATRON (Stach et al., 2018b), users express their privacy requirements in natural language to domain experts, i. e., which knowledge must not be revealed to a third party. The domain experts then analyze the IoT setting (e. g., available data sources or data consumers) and derive privacy threats. In addition, they determine from which data the knowledge that has to be concealed can be obtained. Based on this preparatory work, they create the private patterns, which are then converted into a configuration for PATRON as described in Section 6.2. Users can also formulate service quality requirements, which are similarly converted into public patterns. To reduce manual effort, PATRON introduces tool support to assist experts in identifying privacy threats (Mindermann et al., 2017).

ACCESSORS (Stach and Mitschang, 2018a) provides an additional support for the experts. Using this modeling technique, experts can describe correlations between data sources and derivable knowledge. This model is used for further analyses. To this end, the *RAKE* algorithm initially extracts keywords from the privacy requirements. These keywords are then correlated with the derivable information modeled in ACCESSORS to identify privacy threatening data sources. Since this can be done automatically, domain experts only have to intervene manually if the

model has to be extended, e. g., when new data sources are available. Using *collaborative filtering*, it is also possible to recommend further privacy requirements that the user did not think about. *EPICUREAN* (Stach and Steimle, 2019) is such a recommender system for privacy requirements.

PSSST! makes use of all those tools. Figure 7 depicts the process of how a PSSST! configuration is generated. Similar to PATRON, users describe their privacy requirements in natural language. Smart Service Platforms can provide their requirements towards data quality as well—i. e., they disclose partially which patterns they need to detect in order to provide a certain service. The public and private patterns are then derived from this information using PATRON's tool support as well as EPICUREAN's machine learning techniques. The therefore applied machine learning models can be trained in a privacy-aware manner (Shu et al., 2018). Subsequently, these patterns are deployed in the Secure Access Control Layer of each relevant Smart Environment. There, the quality metric determines which configuration matches these patterns best.

Stach et al. (Stach et al., 2018a) discuss how configurations for a privacy system can be deployed in heterogeneous infrastructures whereas Giebler et al. (Giebler et al., 2018) introduce an architecture to manage both data streams and databases via a shared configuration. Since different Smart Environments are also very heterogeneous and have to deal with both real-time data (via data stream processing systems) and history data (via database systems), these approaches can be used in PSSST! to distribute the configuration.

7 DISCUSSION

In the following, we assess whether the concept and implementation strategy of PSSST! meets all requirements towards a privacy system for Smart Service Platforms (see Section 3) and thus eliminates all threats caused by such a platform (see Section 2.3).

By extending the adapter concept and integrating the techniques from the PMP Resources, data access can be restricted directly at the sensor. The data can either be completely suppressed or manipulated before they are forwarded ($\mathbf{R_1}$). Via the Secure Access Control Layer, PSSST! is able to monitor and control all data sources available in a Smart Environment. Additionally, the VDSs in these layers contain the complete data history. As all of the layer instances are configured and coordinated by a central control unit, PSSST! has total control over all data available to a Smart Service Platform ($\mathbf{R_2}$). Similar to the sensors, also actuators can be secured by applying the concept of PMP Resources to their adapters. In this way, their functionality can be restricted ($\mathbf{R_3}$). CHARIOT's asymmetric encryption can be used for both protecting communication channels and authenticating sensors and actuators. Thus, third parties can neither intercept data nor inject corrupted data into the system ($\mathbf{R_4}$ & $\mathbf{R_5}$). All sensor data must be obtained in PSSST! via the adapters. These adapters are designed for the respective sensor, i. e., they can verify the data and handle sensor errors ($\mathbf{R_6}$).

The requirements that are not directly related to privacy issues are addressed in PSSST! as well. While private patterns are used to ensure privacy, PSSST! uses public patterns to maximize the quality of Smart Services. Its fine-granular access control impairs data quality as little as possible, so that even unspecified public patterns can still be detected ($\mathbf{R_7}$). The adapter concept enables PSSST! to be extendable as required. Therefore, it dynamically adapts to structural and technological changes in Smart Environments (e. g., adding a new sensor) at any time. ACCESSORS enables experts to model new findings regarding the relationships between data and derivable knowledge and to apply them directly to existing and new Smart Services. Thus, PSSST! is also extendable with regard to new Smart Services ($\mathbf{R_8}$). Finally, in PSSST! a user defines his or her privacy requirements at a high level in natural language. As a result, the configuration is user-friendly and the privacy settings exactly meet his or her expectations. Additionally, EPICUREAN identifies other relevant privacy requirements, whereby the configuration is further simplified ($\mathbf{R_9}$).

Thus, PSSST! meets all requirements towards a privacy system for Smart Service Platforms and offers a comprehensive privacy protection for such platforms.

8 CONCLUSION

Smart Services powered by IoT technologies are becoming increasingly popular. They are applied in various domains to facilitate our everyday life. Smart Service Platforms such as Alexa collect all kinds of sensor data and process these data in order to be able to derive the current situation. Thereby they are able to tailor their services to their users and even to react autonomously to certain situations with the help of actuators. An example of such a service is the user-specific control of a Smart Heating.

However, these Smart Services also represent a serious privacy threat, as they collect and process a great amount of data on the one hand and on the other, they are able to operate autonomously. In this paper we therefore examined a real-world application example

for Smart Services and analyzed the functionality of a Smart Service Platform. Based on these insights, we were able to identify privacy threats related to Smart Services. To address these threats, we specified requirements towards a privacy system appropriate for Smart Service Platforms. As state-of-the-art privacy approaches do not meet these requirements properly (i. e., they are overly restrictive, they overburden the users, or they are not applicable to Smart Services), we came up with a novel concept for a novel **p**rivacy **s**ystem for **S**mart **S**ervice Pla**t**forms called *PSSST!*. For the realization of PSSST! we combined three existing privacy systems, namely the PMP, PATRON, and CHARIOT. The final evaluation of PSSST! confirms that our approach meets all requirements.

ACKNOWLEDGEMENTS

We thank the BW-Stiftung for financing both, the PATRON as well as the MIALinx research project and the DFG for funding the SitOPT research project.

REFERENCES

Alhadlaq, A., Tang, J., Almaymoni, M., and Korolova, A. (2017). Privacy in the Amazon Alexa Skills Ecosystem. In *Proceedings of the 17th Privacy Enhancing Technologies Symposium*, pages 1–2.

Alpers, S., Betz, S., Fritsch, A., Oberweis, A., Schiefer, G., and Wagner, M. (2018). Citizen Empowerment by a Technical Approach for Privacy Enforcement. In *Proceedings of the 8th International Conference on Cloud Computing and Services Science*, pages 589–595.

Apthorpe, N., Reisman, D., and Feamster, N. (2016). A Smart Home is No Castle: Privacy Vulnerabilities of Encrypted IoT Traffic. In *Proceedings of the 2016 Workshop on Data and Algorithmic Transparency*, pages 1–6.

Barth, S. and de Jong, M. D. T. (2017). The privacy paradox – Investigating discrepancies between expressed privacy concerns and actual online behavior – A systematic literature review. *Telematics and Informatics*, 34(7):1038–1058.

Beresford, A. R., Rice, A., Skehin, N., and Sohan, R. (2011). MockDroid: Trading Privacy for Application Functionality on Smartphones. In *Proceedings of the 12th Workshop on Mobile Computing Systems and Applications*, pages 49–54.

Cao, J., Carminati, B., Ferrari, E., and Tan, K.-L. (2009). ACStream: Enforcing Access Control over Data Streams. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, pages 1495–1498.

Chung, H., Iorga, M., Voas, J., and Lee, S. (2017a). "Alexa, Can I Trust You?". *Computer*, 50(9):100–104.

Chung, H., Park, J., and Lee, S. (2017b). Digital forensic approaches for Amazon Alexa ecosystem. *Digital Investigation*, 22(Supplement):S15–S25.

Conti, M., Nguyen, V. T. N., and Crispo, B. (2010). CRePE: Context-related Policy Enforcement for Android. In *Proceedings of the 13th International Conference on Information Security*, pages 331–345.

Dwork, C. (2006). Differential Privacy. In *Proceedings of the 33rd International Conference on Automata, Languages and Programming*, pages 1–12.

Felt, A. P., Egelman, S., and Wagner, D. (2012a). 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*, pages 33–44.

Felt, A. P., Ha, E., Egelman, S., Haney, A., Chin, E., and Wagner, D. (2012b). Android Permissions: User Attention, Comprehension, and Behavior. In *Proceedings of the Eighth Symposium on Usable Privacy and Security*, pages 3:1–3:14.

Giebler, C., Stach, C., Schwarz, H., and Mitschang, B. (2018). BRAID — A Hybrid Processing Architecture for Big Data. In *Proceedings of the 7th International Conference on Data Science, Technology and Applications*, pages 294–301.

Gritti, C., Önen, M., and Molva, R. (2018). CHARIOT: Cloud-Assisted Access Control for the Internet of Things. In *Proceedings of the 16th International Conference on Privacy, Security and Trust*, pages 1–5.

Gritti, C., Önen, M., and Molva, R. (2019). Privacy-preserving delegatable authentication in the Internet of Things. In *Proceedings of the 34th ACM/SIGAPP Symposium On Applied Computing*, pages 1–8.

Hornyack, P., Han, S., Jung, J., Schechter, S., and Wetherall, D. (2011). These Aren't the Droids You're Looking for: Retrofitting Android to Protect Data from Imperious Applications. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, pages 639–652.

Jeon, J., Micinski, K. K., Vaughan, J. A., Fogel, A., Reddy, N., Foster, J. S., and Millstein, T. (2012). Dr. Android and Mr. Hide: Fine-grained Permissions in Android Applications. In *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, pages 3–14.

Johnson, N., Near, J. P., and Song, D. (2018). Towards Practical Differential Privacy for SQL Queries. *Proceedings of the VLDB Endowment*, 11(5):526–539.

Kabra, G., Ramamurthy, R., and Sudarshan, S. (2006). Redundancy and Information Leakage in Fine-grained Access Control. In *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pages 133–144.

Kleiminger, W., Mattern, F., and Santini, S. (2014). Predicting household occupancy for smart heating control: A comparative performance analysis of state-of-the-art approaches. *Energy and Buildings*, 85:493–505.

Kotecha, R. and Garg, S. (2017). Preserving output-privacy in data stream classification. *Progress in Artificial Intelligence*, 6(2):87–104.

Migliavacca, M., Papagiannis, I., Eyers, D. M., Shand, B., Bacon, J., and Pietzuch, P. (2010). DEFCON: High-performance Event Processing with Information Security. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, pages 1–15.

Mindermann, K., Riedel, F., Abdulkhaleq, A., Stach, C., and Wagner, S. (2017). Exploratory Study of the Privacy Extension for System Theoretic Process Analysis (STPA-Priv) to elicit Privacy Risks in eHealth. In *Proceedings of the 2017 IEEE 4th International Workshop on Evolving Security & Privacy Requirements Engineering*, pages 90–96.

Mood, B. and Butler, K. R. B. (2018). PAL: A pseudo assembly language for optimizing secure function evaluation in mobile devices. *Journal of Information Security and Applications*, 40:78–91.

Orr, D. A. and Sanchez, L. (2018). Alexa, did you get that? Determining the evidentiary value of data stored by the Amazon Echo. *Digital Investigation*, 24:72–78.

Palanisamy, S. M., Dürr, F., Tariq, M. A., and Rothermel, K. (2018). 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*, pages 40–51.

Purington, A., Taft, J. G., Sannon, S., Bazarova, N. N., and Taylor, S. H. (2017). "Alexa is My New BFF": Social Roles, User Satisfaction, and Personification of the Amazon Echo. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pages 2853–2859.

Rashidi, P. and Cook, D. J. (2009). Keeping the Resident in the Loop: Adapting the Smart Home to the User. *IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans*, 39(5):949–959.

Samsung Newsroom (2018). Samsung Delivers Vision for Open and Intelligent IoT Experiences to Simplify Everyday Life. Press release, Samsung. https://news.samsung.com/us/samsung-vision-iot-experiences-ces2018-press-conference.

Schaub, F., Balebako, R., and Cranor, L. F. (2017). Designing Effective Privacy Notices and Controls. *IEEE Internet Computing*, 21(3):70–77.

Shu, J., Jia, X., Yang, K., and Wang, H. (2018). Privacy-Preserving Task Recommendation Services for Crowdsourcing. *IEEE Transactions on Services Computing*, pages 1–13.

Sopaoglu, U. and Abul, O. (2017). A Top-Down k-Anonymization Implementation for Apache Spark. In *Proceedings of the 2017 IEEE International Conference on Big Data*, pages 4513–4521.

Stach, C. (2018). Big Brother is Smart Watching You: Privacy Concerns about Health and Fitness Applications. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, pages 13–23.

Stach, C., Alpers, S., Betz, S., Dürr, F., Fritsch, A., Mindermann, K., Palanisamy, S. M., Schiefer, G., Wagner, M., Mitschang, B., Oberweis, A., and Wagner, S. (2018a). 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*, pages 372–379.

Stach, C., Dürr, F., Mindermann, K., Palanisamy, S. M., and Wagner, S. (2018b). How a Pattern-based Privacy System Contributes to Improve Context Recognition. In *Proceedings of the 2018 IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 238–243.

Stach, C. and Mitschang, B. (2013). Privacy Management for Mobile Platforms – A Review of Concepts and Approaches. In *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management*, pages 305–313.

Stach, C. and Mitschang, B. (2018a). ACCESSORS: A Data-Centric Permission Model for the Internet of Things. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy*, pages 30–40.

Stach, C. and Mitschang, B. (2018b). 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 ACM/SIGAPP Symposium On Applied Computing*, pages 533–540.

Stach, C. and Steimle, F. (2019). 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 ACM/SIGAPP Symposium On Applied Computing*, pages 1–8.

Stach, C., Steimle, F., and Mitschang, B. (2018c). THOR — Ein Datenschutzkonzept für die Industrie 4.0: Datenschutzsysteme für die Smart Factory zur Realisierung der DSGVO. In *Workshops der INFORMATIK 2018*, pages 71–83.

Townsend, D., Knoefel, F., and Goubran, R. (2011). Privacy versus autonomy: A tradeoff model for smart home monitoring technologies. In *Proceedings of the 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 4749–4752.

Wallis, K., Hüffmeyer, M., Koca, A. S., and Reich, C. (2018). Access Rules Enhanced by Dynamic IIoT Context. In *Proceedings of the 3rd International Conference on Internet of Things, Big Data and Security*, pages 204–211.

Wieland, M., Hirmer, P., Steimle, F., Gröger, C., Mitschang, B., Rehder, E., Lucke, D., Rahman, O. A., and Bauernhansl, T. (2016). Towards a Rule-based Manufacturing Integration Assistant. *Procedia CIRP*, 57(1):213–218.

Zhang, J., Li, H., Liu, X., Luo, Y., Chen, F., Wang, H., and Chang, L. (2017). On Efficient and Robust Anonymization for Privacy Protection on Massive Streaming Categorical Information. *IEEE Transactions on Dependable and Secure Computing*, 14(5):507–520.

Zheng, S., Apthorpe, N., Chetty, M., and Feamster, N. (2018). User Perceptions of Smart Home IoT Privacy. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):200:1–200:20.