The Privacy Management Platform
An Enabler for Device Interoperability and Information Security in mHealth Applications

Christoph Stach, Frank Steimle and Bernhard Mitschang
Institute for Parallel and Distributed Systems, University of Stuttgart, Universitätsstraße 38, D-70569 Stuttgart, Germany

Keywords: mHealth, Device Interoperability, Information Security, COPD, Privacy Management Platform.

Abstract: Chronic diseases are on the rise. Afflicted patients require persistent therapy and periodic screenings. This causes high treatment costs and overburdened physicians. Innovative approaches that enable patients to perform treatment methods on their own are badly needed. Telemedical approaches with the aid of modern Smartphones connected to medical devices (the so-called mHealth) can be the answer. However, mHealth apps face two key challenges, namely device interoperability and information security. In this paper, we describe how the **P**rivacy **M**anagement **P**latform (*PMP*) and its extendable *Resources* can contribute to these challenges. Therefore, we analyze a real-world mHealth app and derive generic functional units, each realizing a certain task recurring frequently within mHealth apps, e. g., metering, data storage, or data transmission. For each functional unit we provide a PMP Resource, enabling both, device interoperability and information security. Finally, we revise the analyzed mHealth app using the Resources in order to evaluate our approach.

1 INTRODUCTION

Due to long stand-by times and multiple built-in sensors, the Smartphone became our ubiquitous companion. New use cases are constantly emerging. Especially in the health sector, the use of Smartphones can be highly beneficial to save treatment costs and help patients who cannot visit their physicians regularly (Silva et al., 2015). The usability of Smartphone apps[1] in the health sector—the so-called mHealth apps—is limitless. There is an app for almost any situation (Siewiorek, 2012).

While there are some mHealth apps for medical reference (that is, apps providing information about diseases) as well as hospital workflow management apps (i. e., apps supporting physicians in their everyday duties), mHealth apps are mainly from the health management domain (Milošević et al., 2011). The latter includes cardio fitness, medication adherence, and chronic disease management. To this end, these apps support two essential features: self-observation and feedback (Mattila et al., 2010). That is, the patient performs health measurements instructed by the app, transmits the measured values to the app and the app performs analyses on these values. Based on the results, the app gives the user medical recommendations.

Since mHealth apps involve the patient actively into the treatment and monitoring process, s/he gets more aware of his or her condition. So, mHealth apps change the physician patient relationship especially for patients with a chronic disease. These patients have to visit their physician periodically in order to check certain health values. However, such a metering can be performed by the patients autonomously, if they receive a proper guidance tailored to their know-how. That is, with the help of an mHealth app they are able to do the monitoring in a telemedical manner. This take the load off both, patients as well as physicians. Patients benefit from the freedom to do the metering at any arbitrary place and time while physicians are able to concentrate on emergencies. This is not just a huge saving potential but also improves the quality of healthcare at the same time (Schweitzer and Synowiec, 2012).

To capture health data, mHealth apps make use of various sensors. In addition to the already broad spectrum of sensors built into modern Smartphones that can be used for healthcare (e. g., a heart rate sensor, a camera, or a microphone), even medical devices for home-use can be connected to a Smartphone. The

[1] In the following, we use the generic term *app* for any kind of mobile application.

Stach, C., Steimle, F. and Mitschang, B.
The Privacy Management Platform - An Enabler for Device Interoperability and Information Security in mHealth Applications.
DOI: 10.5220/0006537300270038
In *Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2018) - Volume 5: HEALTHINF*, pages 27-38
ISBN: 978-989-758-281-3

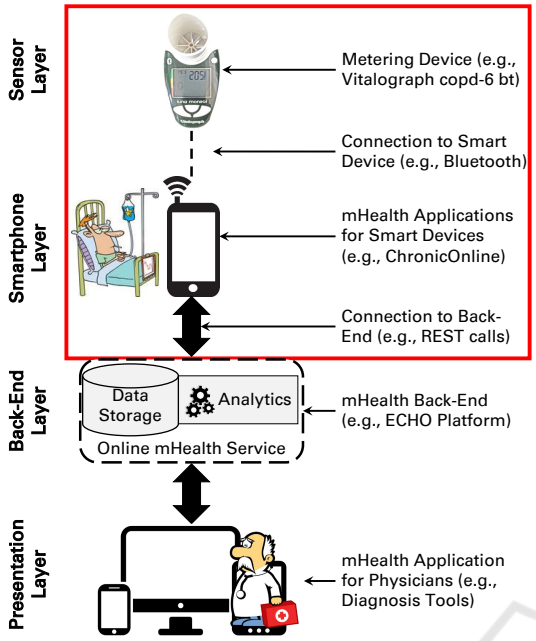


Figure 1: Overview of an mHealth System (Red-Rimmed Components are in the Focus of this Paper).

Vitalograph copd-6 bt is an example for such medical device.

Yet, the data interchange between medical devices and Smartphones often fails because of non-uniform communication protocols. The **device interoperability** of Smartphones and medical devices is a key challenge for the success of mHealth apps (Chan et al., 2012). Also the assurance of **information security** is vital for mHealth, as patients have to trust their apps (Bai et al., 2014). Thus, we address these two challenges in our work. Therefore, we come up with a concept for an enabler for device interoperability and information security in mHealth apps. To achieve this objective, we proceed as follows: (*I*) We analyze a real-world mHealth app regarding the collected data and used devices. (*II*) We deduce a generic data model for mHealth apps. Due to this data model, our approach is applicable for any kind of mHealth app. (*III*) We identify five recurring tasks within mHealth apps for which device interoperability and information security are key requirements, namely login, metering, localization, data storage and analytics, and data transmission. For each of these tasks, we introduce an extension for the **P**rivacy **M**anagement **P**latform (*PMP*) (Stach and Mitschang, 2013), to ensure interoperability and information security. (*IV*) We use these extensions to revise the analyzed mHealth app in terms of sensor support and the patients' privacy. With this revised app, we assess the practical effect of our approach.

The remainder of this paper is as follows: Section 2 introduces a real-world application scenario from the mHealth domain, namely ChronicOnline, an mHealth app for COPD patients. Then, Section 3 takes a look at related work concerning connection techniques used by medical device as well as information security mechanisms in mHealth apps. Based on these findings, we introduce a generic interchange data model for mHealth apps and induct briefly in the PMP, the foundation platform for our solution approach in Section 4. Then, Section 5 details on the key components of our approach and demonstrates their applicability, using the example of a COPD app. In Section 6 we assess how our approach contributes to solve information security and interoperability problems for mHealth apps. Section 7 concludes this work and gives an outlook on some future work.

2 APPLICATION SCENARIO

Typically, mHealth systems consist of three layers (Kumar et al., 2013). The *Sensor Layer* manages the access to any sensor required by mHealth apps, i. e., it collects the sensors' raw data and provides it to subsequent layers. In the *Smartphone Layer*, this data is collected, assessed, and processed. The processing step transforms the raw data to information. An optional analysis phase can derive events from the information, e. g., to detect a seizure automatically. The data can also be stored on the Smartphone, e. g., to monitor the progress of the disease. Additionally, the data is forwarded to on-line servers managed by the *Back-End Layer*. While the Smartphone Layer holds health data of a single patient, in the Back-End Layer the data of multiple patients is assembled (e. g., to derive new insights into the course of a disease or to prepare the data for subsequent in-depth analyses). Via this layer physicians are able to perform analyses and receive diagnosis support for each of their patients. For this purpose, there is sometimes an additional *Presentation Layer*. This fourth layer preprocesses the health data stored in the back-end and presents the relevant data in a user-friendly manner. So, physicians are able to interpret the results.

Figure 1 shows the interaction of these four layers. A metering device can be connected to a Smartphone via Bluetooth. A patient can install an mHealth app on his or her Smartphone and use the medical data recorded by the external metering device. This data can be enriched by data from the Smartphone's built-in sensors (e. g., location data). The app sends the gathered data to an mHealth back-end for thorough analyses and to store the data at a central repository.

Physicians can access the repository via their diagnosis tools to find an adequate method of treatment.

Concerning the two key issues addressed in this paper, namely device interoperability and information security, several components have to be taken into consideration. Any data interchange between two layers is a problem, since there are heterogeneous interchange formats and multiple connection standard. This concerns especially the data interchange between Smartphones and medical devices, since these devices commonly define proprietary communication protocols. Yet, the harmonization of the medical data formats used by mHealth apps and back-ends has to be considered as well. While there are several approaches towards a server-sided unified data model for health data (e. g., in the *HealthVault* (Bhandari, 2012)), there are no such approaches for apps.

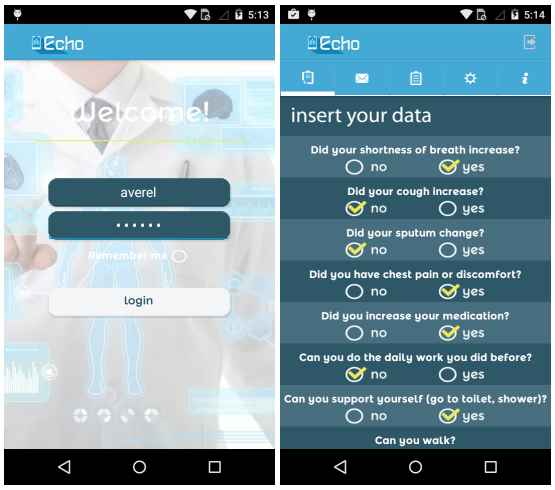
Information security has to be considered at any layer. However, as users cannot influence the security mechanisms implemented in sensors, data protection has to be assured mainly on the Smartphone Layer. While there are approaches to protect sensitive data on the Back-End Layer (e. g., (Jafari et al., 2011)), an adequate solution for the Smartphone Layer is missing. The Presentation Layer does not create any sensitive data.

So, we focus on the Sensor Layer and the Smartphone Layer in our work regarding both, device interoperability and information security, as there are adequate solutions for the other layers solving theses issues. However, the usability and security of an mHealth system is impaired by its weakest component. That is, the existing usability and security solutions for the Back-End Layer and the Presentation Layer are worthless with respect to the whole system, as long as there are no appropriate approaches for the Sensor Layer and the Smartphone Layer as well.

In the following, we introduce a real-world mHealth app for COPD patients.

Chronic Obstructive Pulmonary Disease. COPD is an obstructive lung disease. Patients suffer from a poor airflow which worsens over time. According to the WHO, approximately 6% of all deaths in 2012 resulted directly from COPD. Even though COPD is not curable, a fast and persistent therapy can slow the progression of the disease significantly down. Telemedicine is very appropriate as the required measurements can be carried out with medical metering devices for home use (de Toledo et al., 2006).

The ChronicOnline App. mHealth apps are good for regularly recording various parameters but they are insufficient for a comprehensive COPD screening. For this reason, the University of Stuttgart, the University of Crete, and OpenIT launched the *ECHO* project in



(a) Login Screen (b) Questionnaire

Figure 2: The ChronicOnline App's Key Functionality.

2013[2]. Within the scope of this project several mHealth apps collect various health data and gather the data in a Cloud infrastructure. Online services are available for physicians enabling various data analytic functions and giving them an holistic overview of their patients' condition. In addition to it, the patients and the physicians remain in contact with each other whereby the physicians can give their patients advices by sending them messages via their mHealth app. The *ChronicOnline* app (Bitsaki et al., 2016) is a mobile front-end for the ECHO project. Its key functions (user management and a COPD questionnaire) are shown in Figure 2.

Initially, the user has to log in his or her account (see Figure 2a). The app differentiates two user groups, patients who have only access to their personal account and physicians who monitor several patients. The login process is realized by REST calls[3] to the ECHO back-end.

After authorization is complete, the patient has access to several tabs. The most significant tab is the questionnaire tab (see Figure 2b). Here the user has to answer five questions about his or her condition. Each question can be answered with 'yes' or 'no'. Depending on the given answers up to six subquestions appear to refine the medical finding. Afterwards, the results are transferred to the back-end as a JSON object. More details on the data format are given in Section 4.1.

The back-end performs analyses and preprocesses the data for physicians. The physicians can contact their patients via the *Amazon Simple Notification Ser-*

[2]see http://chroniconline.eu

[3]see (Fielding, 2000)

vice (*SNS*)[4]. These messages are meant for vital issues (e. g., "Go to the Hospital!"). The discussion of the medical findings in case of a aggravation still has to take place in a personal meeting with the physician.

As for both, physicians and patients, the progression of the health condition is an important information, the app can retrieve history data from the back-end (e. g., to display former questionnaire answers). For detailed analyses and history charts, especially for the physicians, the input and output capabilities of a mobile device are less applicable, which is why currently such profound reports are stripped off the app completely.

This app is a representative sample for the innumerable COPD apps available in various app stores. We could use any of them without a loss of argument. Please note that the back-end is out of this paper's scope. More information about the back-end can be found in (Steimle et al., 2017). However, we give a brief outlook on a information security mechanism for the processing of sensitive data in such a back-end in Section 7.

Applied Metering Devices. The Vitalograph copd-6 bt[5] is used to screen the pulmonary function. So, patients with respiratory conditions can perform the metering by themselves. It records various lung function parameters including the Peak Expiratory Flow (PEF) and the Forced Expiratory Volume (FEV) among others. PEF and FEV are key readings for the diagnosis of COPD. The measurement results are transmitted via Bluetooth LE. As today's Smartphones support this battery-saving connection standard, the Vitalograph copd-6 bt provides a sound hardware foundation for telemedical mHealth apps.

For data interchange the proprietary *Terminal I/O* protocol[6] is used which operates on top of Bluetooth LE GATT[7]. The basic idea behind this protocol is that a client (e. g., a Smartphone) has to request credits from the server (i. e., the Vitalograph copd-6 bt). For each message the client has to pay credits. So, the client is able to make a specific number of requests without having to wait for a response from the server. The application of such proprietary protocols impedes the development of mHealth apps since only a limited number of devices supports the respective protocol. As a common communication standard is not in sight, app developers are in great need of other approaches that enhance device interoperability.

[4]see https://aws.amazon.com/sns

[5]see https://vitalograph.com/product/162427

[6]see (Stollmann Entwicklungs- und Vertriebs-GmbH, 2014)

[7]see (Bluetooth SIG, Inc., 2017)

3 RELATED WORK

In the context of device interoperability in mHealth apps, a lot of work is done concerning the back-end systems. For this purpose, these systems introduced harmonized data models for health data and provide generic interfaces so that any kind of mHealth app can use them to collect and share heath data. One of the biggest systems is the HealthVault (Bhandari, 2012). It supports various health-related data types ranging from fitness data to entire personal health records. The HealthVault acts as a middleman between the data producers—i. e., the mHealth apps—and the data consumers—i. e., analysis systems. Concerning app-sided device interoperability and information security, such a system provides no help. *Google Fit* (Mishra, 2015) is another back-end for storing and processing health data. Google's system deals with fitness data (e. g., the heart rate), only. Google Fit provides interfaces for app developers which enhance the device interoperability and facilitate the reading of sensors in third-party devices—at least for devices that are supported by Google Wear[8]. As especially medical devices are not supported by Google Wear, (O'Donoghue and Herbert, 2012) discuss, how Smartphones can be connected with these devices. However, their solution aims for physical connections and not for harmonized communication standards.

(Kouris and Koutsouris, 2014) recommend to use IoT techniques to solve this problem. In their proposal all sensors are connected to the Internet and send their data to a Cloud-based database which is also accessible for mHealth apps. Even though there are secure transfer protocols for health data (e. g., (Mare et al., 2014)), a permanent and unrestricted transmission of such sensitive data to an unknown server is ineligible for most users. For that reason, (Gardner et al., 2009) introduce an approach with which the patient stays in total control over his or her data; the mHealth app has full access to the health data while external services (e. g., apps for physicians) only get access as long as the patient grants it. In (Murad et al., 2013) an mHealth app is introduced which relies on a full encryption of the health data when the data is stored or transmitted. Yet, both approaches assume that only external entities constitute a threat for the security of health data. However, as studies prove, two out of three apps handle their data either carelessly or even maliciously (Enck et al., 2010).

Thus, none of these approaches solves the device interoperability and information security issues of mHealth apps. Despite the benefit of mHealth apps especially the device incompatibility and the mis-

[8]see https://www.android.com/wear

```
DailyReport {
  patientId (int): Patient ID,
  recordId (int): Record ID,
  date (string): Date of Report,
  q1 (boolean): Answer to Question 1,
  ...
}
```

Listing 1: The ChronicOnline Data Model (excerpt).

trust in app developers repel patients from using such apps (Murnane et al., 2015). For this reason, we introduce a general approach dealing with both issues in the following sections.

4 INTEROPERABILITY AND SECURITY REFLECTIONS

In order to come up with a design methodology for interoperable and secure mHealth apps, a secure data management, data input techniques, and defined data access conditions are required. Our solution to this is built upon the PMP. When looking at the ChronicOnline app, it does not take advantage of the full potential of the ECHO back-end and the prevailing hardware. With its analytic functions and notification services, the back-end is able to process more complex data such as location data or respiratory data in addition to the replies to the disease-specific questionnaire. By the integration of medical devices, the ChronicOnline app becomes a full-fledged telemedicine solution.

Therefore we extend the ChronicOnline app by adding location services[9]. Studies show that the location can have a relevant influence on the progression of a disease (Knöll and Moar, 2011). Moreover, we provide support for various third-party Bluetooth respiratory monitors. The measurement results are added to the electronic health record (see Section 4.1 for its data model) and transferred to the ECHO back-end. There, the data can be automatically pre-analyzed which not only unburdens the physicians in charge, but also results in a faster feedback for the patients. As a consequence this enhanced app has to deal with increasing interoperability and security issues.

Amulet, a tiny Smart Device operating as an information hub, tries to solve both problems (Hester et al., 2016). It confirms the user's identity and identifies any available devices in the surrounding belonging to him or her. Then, Amulet ascertains that only trusted third-party devices can be used for the metering and ensures a secure connection to these devices. Moreo-

[9]Keep in mind, that location based services in general constitute a severe threat to a user's privacy (Marcelino and Silva, 2018).

```
DailyReport {
  ...
  q5 (boolean): Answer to Question 5,
  lat (int): Latitude,
  lon (int): Longitude,
  pef (int): Peak Expiratory Flow,
  fev (int): Forced Expiratory Volume,
  ...
}
```

Listing 2: The Extended Data Model (excerpt).

ver, Amulet provides mechanisms to protect the health data against external attackers. In order to transfer the data to (trusted) servers for further processing, Amulet is able to connect to a Smartphone and use it for transmission. Unfortunately, this approach has a severe drawback: The user has to possess another device in addition to his or her Smartphone and the actual medical device. This causes further costs and it is unpractical since the user has to carry the Amulet all the time. The PMP (see Section 4.2) is a middleware for application platforms which provides similar features.

4.1 The Internal Data Model

The ChronicOnline app sends up to eleven boolean values, representing the answers to the questionnaire, to the ECHO back-end. For this reason, its data model for the data exchange is quite plain (see Listing 1). In addition to the questionnaire answers, a daily report entails an ID for the patient and for the report itself as well as the submission date. Please note that authorization data to confirm the identity of the submitter is not part of this data model. The authorization is managed via the authorization header of the HTTP protocol.

We add latitude and longitude to this basic schema to support location data as well as entries for the most relevant COPD readings, including the Peak Expiratory Flow and the Forced Expiratory Volume among others (see Listing 2).

Since the JSON format is well-suited for the data exchange between the app and the back-end, but not for the data processing within the app, we apply wrapper classes for the conversion of such JSON files to Java objects and vice versa.

4.2 Overview of the PMP

We use the PMP in order to realize interoperability and security features as needed by mHealth apps. We give a brief overview of the PMP at first and describe in detail the new components, which are developed in this work in the next section. The PMP is an intermediate layer between apps and the operating system. It pre-

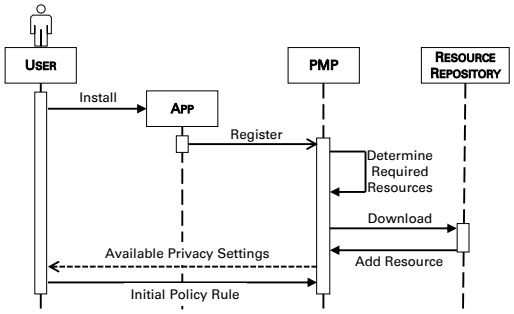


Figure 3: The Registration of an App at the PMP.

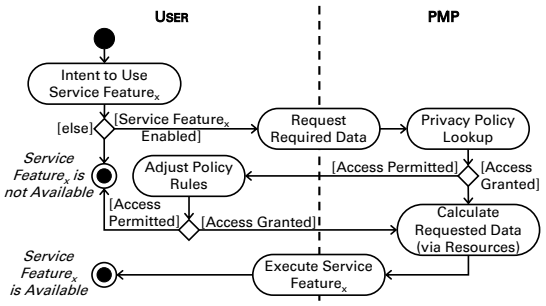


Figure 4: The PMP's Permission Allocation Process.

vents any (potentially malicious) app from accessing sensitive data.

When an app needs access to such data, it has to ask the PMP for permission. The PMP operates several data provisioning services, the so-called *Resources*. Each Resource is responsible for a certain type of data but it is not committed to a certain technology; e. g., the Location Resource is able to provide location data from a GPS provider or from a network provider. Thereby it can adapt its functionality to the available hardware. In addition to it, the user can define how accurate the data should be in order to obfuscate his or her private data. Further Resources can be hooked into the PMP at runtime need-based.

A Resource defines *Privacy Settings* which restrict the usage of the corresponding Resource. By default, there is a Privacy Setting for granting or permitting the usage of a Resource. Furthermore, a Privacy Setting can be more specific depending on the type of Resource. For instance, the Location Resource can reduce the location's accuracy.

At installation time an app has to register at the PMP. The PMP identifies which Resources are required and installs missing Resources if necessary. The user then postulates an initial policy rule for this app defining, which data should be given to the app and how accurate this data should be. This registration process is shown in Figure 3.

Since the user is able to deny that an app gets access to a certain Resource, the app model of the PMP encapsulates logically coherent parts of the app in so-called *Service Features*. So, the withdrawal of access rights simply deactivates the affected Service Features but the app itself can still be executed. Moreover, the user can modify access rights of individual Service Features at runtime. The permission allocation is shown in Figure 4.

The PMP is primary a fine-grained permission system with additional privacy features (e. g., data obfuscation). However, in the context of interoperability and uncertainty of available hardware, the PMP serves a dual purpose. Each Resource is abstracted from a certain technology and can have several implementations. As a consequence, the app developer only has to request a certain type of data (e. g., respiratory data) and the dedicated Resource ensures that it gets this data from the available hardware (e. g., a Vitalograph copd-6 bt). If no hardware providing this kind of data is available—which is similar to a user-defined prohibition to use the hardware via the PMP—then the app gets informed and adapts to this condition by deactivating affected Service Features.

The PMP is able to degrade an app's functionality when it cannot access all of its requested data. Services Features can be deactivate instead of feeding an app with random data—in the context of mHealth apps, the usage of random medical values is inappropriate. For this reason, the PMP's data obfuscation for health data is severely restricted. For further details on the PMP, please refer to the literature (Stach and Mitschang, 2013, Stach, 2013, Stach and Mitschang, 2014, Stach, 2015).

5 DESIGN OF MHEALTH PMP RESOURCES

In the following we introduce five PMP Resources which are developed for mHealth apps, namely a secure **authentication** Resource, a **metering** Resource, a **localization** Resource, a **data storage** Resource, and a **connection** Resource. In addition, a Resource for health data **encryption** is introduced. Finally, we revise the ChronicOnline app with the help of these Resources.

5.1 The Dialog Box Resource

In the ChronicOnline app a user has to enter credential information which is sent to an ECHO server for verification. While the back-end is operated by a trusted organization (e. g., a hospital), any developer can implement apps for ECHO. Thus the front-end is po-

tentially insecure—yet the user has to reveal his or her login data to it.

In order to solve this problem, we introduce an isolated *Dialog Box Resource*. An app can invoke the dialog box and specify the displayed text as well as where the entered information should be forwarded to. The server's response is sent back to the invoking app. In this way, the dialog box is completely generic and can be used in any context where the user has to enter private data. The dialog box is executed as a part of the PMP and is completely isolated from the invoking app. No information is passed to the app except for the back-end's reply. The user cannot only grant or permit the usage of the dialog box, but also specify which back-ends are legit recipients.

5.2 The Metering Resource

One of the biggest problems for mHealth apps is the integration of third-party medical devices as there is currently no uniform standard for intercommunication with such devices. This is why most of the currently available apps support some hand-picked medical device only.

That is why we introduce a *Metering Resource* with a simple, yet generic interface. Since most modern medical devices exchange their data with Smartphones via a Bluetooth connection, we focus our work on this kind of connection. As there is no common Bluetooth transmission protocol, the Metering Resource has to be able to support several protocols. Currently, the Terminal I/O protocol is supported. However, further protocols (e. g., Android Wear) or other connection standards can be supported in the future due to the modular expandability of PMP Resources. The Metering Resource defines no fixed schema for health data, but processes this data as a JSON object with an attribute for every measured value.

As the Metering Resource autonomously connects to any available device, an app developer simply has to request the health record from the Resource. However, concerning information security, the app should not be able to read the health record. Thus, the PMP encrypts the JSON object containing the health data. As a consequence, authorized Resources can process the health data while (potentially malicious) apps cannot access the data. Nevertheless, sometimes an app needs access to certain values of the health data, e. g., in order to display the data. For this purpose, the *Unsealer Resource* (see Section 5.6) can be used in order to decrypt certain excerpts of the record.

The user is able to restrict which devices are allowed to provide health data for a certain app. For instance, s/he can permit an app to use the Metering Resource, but only a certain kind of data from a specific device is sent back by the Resource.

5.3 The Location Resource

Since the location of a reading can be relevant, we introduce a special *Location Resource* for mHealth apps. Normally, in Android an app has to subscribe for location updates and the system sends any update to the app. This causes a high battery drain. However, such a behavior is not necessary in the context of a mHealth app which requires location data only once after the metering is executed. As a consequence the Location Resource supports two modes of operation: An app can request periodical location updates or only a single location. The Resource arranges the (un)subscription automatically. Android supports several location providers, e. g., a Wi-Fi-based or a GPS. Our Location Resource requests the most accurate provider which is currently available.

In order to respect the user's privacy, the Location Resource provides a Privacy Setting to reduce the accuracy of the location data. The user sets a maximum accuracy in meters. If the provided location data is more accurate, the Resource adds a random factor depending on the user settings to the latitude and longitude in order to reduce the accuracy. The user is also able to use completely randomized or mocked location data.

5.4 The Secure Database Resource

Since Android stores its data in a clear-text readable form, attackers may harvest all stored data. The file system encryption is no solution for this threat, since the data is decrypted as soon as the system is fully booted. So, sensitive data such as health data requires additional security features. The **S**ecure **D**ata **C**ontainer (*SDC*) is a *Secure Database Resource* for the PMP (Stach and Mitschang, 2016). It encrypts the stored data with AES-256 encryption and only the PMP has the key. The SDC has a fine-grained access control to share the stored data with selected apps. Performance measurements show that the thereby caused overhead is within reasonable limits.

We tailor the SDC slightly to our requirements: The database's internal data model is comparable to a key-value store. The Secure Database Resource operates with JSON objects and adopts the therein defined keys and values directly. As several stored JSON objects can use a common key, an internal id is applied to each object. The database's primary key consists of the internal id and the key from the JSON object. The Database Resource's interface is minimalistic, yet

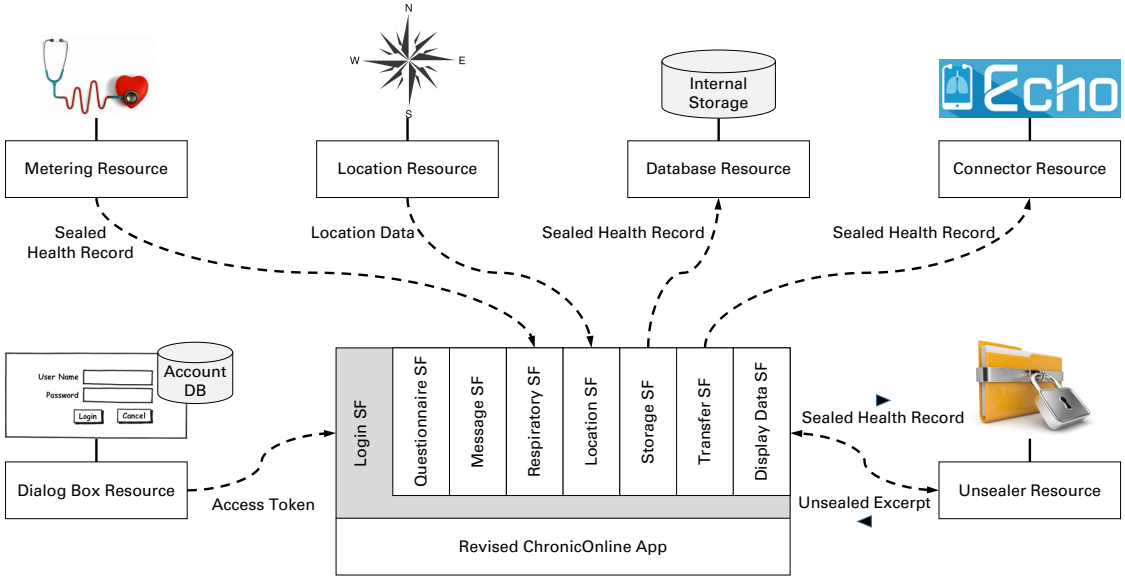


Figure 5: Relations Between the Service Features of the COPD App and the PMP Resources.

demand-actuated. A JSON object can be stored, obtained, and deleted via its key.

5.5 The Connector Resource

Nowadays, most apps do not only operate locally on a single Smartphone, but include various external services. This is why almost every app requests the permission to establish a network connection. The user is not even informed about this request. However, an app having this permission is able to upload sensitive data to any server. To use external services in a secured way, we introduce the *Connector Resource*. Within the Connector Resource various trusted external services are specified. An app is able to either upload data to or retrieve data from one of these services (e. g., the ECHO back-end or Amazon's SNS). Additional domain-specific information protection policies can be applied within the Connector Resource (such as the *Mobile access to Health Documents* profile (Moehrke, 2017b) or the *Audit Trail and Node Authentication* profile (Moehrke, 2017a)), if they are supported by the back-end.

Concerning interoperability, the Connector Resource handles any interface changes of the external services for the apps. That is, adjustments have to be made only at the Resource and not for every app. Additionally, the app's network access is heavily restricted. It is only able to use the Resource's interface and the user can block any external services s/he has no trust in.

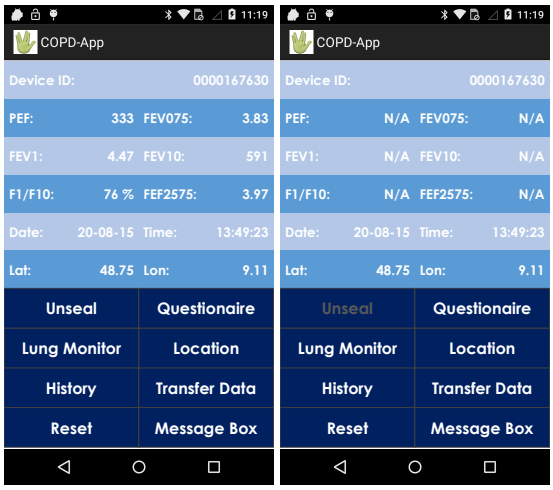
5.6 The Unsealer Resource

As mentioned above, health data gets encrypted by the PMP before it is passed to an app. So, apps cannot access this sensitive data directly. As health data is only processed in Resources, this encryption means no limitation for the user (cf. (Weerasinghe et al., 2009)). If an app still needs access to health data (e. g., to display it), we introduce the *Unsealer Resource* which decrypts selected values. The user is able to define which excerpt of the data should be revealed. To that end, the Resource provides a domain-specific selection (e. g., general information, lung-related content only, etc.) in order to facilitate the configuration.

5.7 Revised ChronicOnline App

Based on these Resources, we revise the ChronicOnline app by including sensor data (e. g., respiratory meters or location sensors) while regarding device interoperability and information security.

After the user has answered the questionnaire, the revised app expects respiratory data from a connected device via the Metering Resource. The user only has to push the "*Lung Monitor*" button and the connection and data transfer is arranged automatically by the PMP. When the measured data is available, the results of the questionnaire are applied to the DailyReport (see Listing 2) and the location data is inquired. Then, the complete DailyReport object is stored internally (e. g., if the data cannot be transmitted immediately) and is transferred subsequently to the ECHO back-end (see Figure 6a). Please note, that for this capture any data



(a) Unsealed Data Screen (b) Sealed Data Screen

Figure 6: The ChronicOnline Privacy-Driven App.

is unsealed for demonstration purpose. The user is informed in each step about the collected data. So, s/he is in total control over the data. The effect of a deactivated Unsealer Resource is shown in Figure 6b. Without the Unsealer Resource only data which are not related to any health issues are known to the app.

In order to react properly to the restriction of a Resource, an app needs to define Service Features. Figure 5 shows the 8 Service Features of the revised ChronicOnline App (denoted as SF) and which Resources are required for each of them. Not every Service Feature necessarily requires data from a Resource (e. g., the *Questionnaire SF*). The Service Features have a modular design and can be plugged in and out at runtime (e. g., when the corresponding Resource is deactivated). In the app's program flow these features are skipped. However, since the authentication of the user is mandatory, the *Login SF* and therewith the Dialog Box Resource cannot be deactivated or else any other Service Feature is also deactivated.

6 ASSESSMENT

Concerning information security, the literature speaks of 7 key protective goals, namely *auditability*, *authenticity*, *availability*, *confidentiality*, *integrity*, *non-repudiation*, and *privacy* (Dhillon and Backhouse, 2000, Cherdantseva and Hilton, 2013). The original ChronicOnline app only fulfills the authenticity goal and the confidentiality goal directly due to its login mechanism and the auditability goal, the integrity goal, and the non-repudiation server-sided due to the security mechanism of the ECHO back-end. However, as soon as real health data is processed by the app, the user cannot rely on the prevailing mechanisms. Our revised app supports all 7 protective goals due to the used Resources. The auditability as well as the non-repudiation is guaranteed, since the PMP logs any Resource access of an app. The authenticity is given via the login mechanism and since the login data is not shared with the app, it cannot commit an identity theft. The availability is given, as all data is stored on the device using mature database technologies. The confidentiality is ensured, since any app functionality is only usable after the login process is completed. The integrity is guaranteed since any relevant data is encrypted and therefore cannot be manipulated by third-parties. Privacy is retained, as the user decides, which data can be used by an app and s/he can specify for any non-health data how accurate or even randomized it should be provided. As an app cannot access any data without using the PMP, data access is strictly constrained by the Resources' interfaces. Thus, from an information security's point of view the revised app satisfies all requirements.

Concerning device interoperability, the modular expandability of the Resources turns out to be beneficial. For instance, support for additional devices can be added to the Metering Resource need-based at runtime. An app developer only has to code against a Resource's interface, no matter which hardware is actually available. Therefore, complex and labor-intensive coding is required only once (for the Resource) and it can be reused many times (in the apps). Additionally, due to their generic design the Resources are usable in many different application scenarios even for non-mHealth apps. Thus, from an interoperability's point of view the PMP Resources satisfy all requirements regarding compatibility and reusability. Table 1 lists the key contributions of the introduced Resources.

7 CONCLUSION AND OUTLOOK

The rising numbers of patients suffering from chronic diseases are responsible for increasing treatment costs as well as overburdened physicians. The usage of a combination of Smartphones with medical devices, commonly referred to as mHealth, can provide a remedy for this situation. Apps realizing the data interchange between the Smartphone and the medical device can be used autonomously by the patients for telemedical screening of their condition. Unfortunately, these mHealth apps face two key challenges, namely information security and device interoperability. Each mHealth app has to reinvent the wheel concerning these two issues. So, the development effort for mHealth apps is unnecessary high and the quality of

Table 1: Feature Summary of the Resources Concerning Device Interoperability and Information Security Issues.

PMP Resource	Device Interoperability	Information Security
Dialog Box Resource	• displayed dialog text can be tailored to the app • evaluation mechanism can be substituted	• entered data is not revealed to the app
Metering Resource	• support of different medical devices • support of different transmission standards	• data is transmitted encrypted • restriction of usable device types
Location Resource	• support of different location sensors • support of different modes of operation	• restriction of location accuracy • use of randomized location data
Database Resource	• generic data model	• full database encryption
Connector Resource	• support of different external services	• no dircet access to the network for the app • restriction of usable services
Unsealer Resource	• all authorized Resources get unencrypted data	• apps have only limited access to health data

information security and device interoperability features differs strongly from app to app. As mHealth apps deal with sensitive data, a consistently high service quality is badly required.

For this reason, we pursue four key objectives with our work: (I) We analyze the ChronicOnline app regarding the collected data and used devices. (II) We deduce a generic data model for mHealth apps from this app. (III) We identify five recurring tasks within mHealth apps for which device interoperability and information security are key requirements. For each of these tasks, we introduce a PMP Resource. Each Resource is designed regarding device interoperability and information security issues. (IV) We use these extensions to revise the ChronicOnline app. With this revised app, we assess the practical effect of our approach. Although our approach is based on Android, the underlying concept can be applied to any application platform.

As the evaluation results in Section 6 show, our approach meets the requirements of mHealth apps. Yet, this is just a first protective measure. Modern Smartphone applications commonly serve as data sources for comprehensive stream processing systems realizing the actual computation. These systems have access to a wide range of sources. With the help of such comprehensive stream processing systems, a lot of knowledge about the patients can be derived. Even if a user restricts access to a certain type of data on his or her device, a stream processing system could be able to retrieve this data from another source. Therefore the privacy rules of each application also have to be applied to affiliated services which process the application's data. Future work has to investigate, how the PMP settings concerning information security can be applied to a privacy mechanism for stream processing systems such as the *PATRON* research project[10] (Stach et al., 2017).

ACKNOWLEDGEMENTS

This paper is part of the PATRON research project which is commissioned by the Baden-Wrttemberg Stiftung gGmbH. The authors would like to thank the BW-Stiftung for the funding of this research.

[10]see http://patronresearch.de

REFERENCES

Bai, Y., Dai, L., and Li, J. (2014). Issues and Challenges in Securing eHealth Systems. *International Journal of E-Health and Medical Communications*, 5(1):1–19.

Bhandari, V. (2012). *Enabling Programmable Self with HealthVault*. O'Reilly Media, Inc., Beijing, Cambridge, Farnham, Köln, Sebastopol, Tokyo.

Bitsaki, M., Koutras, C., Koutras, G., Leymann, F., Steimle, F., Wagner, S., and Wieland, M. (2016). ChronicOnline: Implementing a mHealth solution for monitoring and early alerting in chronic obstructive pulmonary disease. *Health Informatics Journal*, 23(3):197–207.

Bluetooth SIG, Inc. (2017). GATT Specifications. https://www.bluetooth.com/specifications/gatt.

Chan, M., EstèVe, D., Fourniols, J.-Y., Escriba, C., and Campo, E. (2012). Smart Wearable Systems: Current Status and Future Challenges. *Artificial Intelligence in Medicine*, 56(3):137–156.

Cherdantseva, Y. and Hilton, J. (2013). A Reference Model of Information Assurance & Security. In *Proceedings of the 2013 International Conference on Availability, Reliability and Security*, ARES '13, pages 546–555.

de Toledo, P., Jimenez, S., del Pozo, F., Roca, J., Alonso, A., and Hernandez, C. (2006). Telemedicine Experience for Chronic Care in COPD. *IEEE Transactions on Information Technology in Biomedicine*, 10(3):567–573.

Dhillon, G. and Backhouse, J. (2000). Technical Opinion: Information System Security Management in the New Millennium. *Communications of the ACM*, 43(7):125–128.

Enck, W., Gilbert, P., Chun, B.-G., Cox, L. P., Jung, J., McDaniel, P., and Sheth, A. N. (2010). TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI '10, pages 393–407.

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine.

Gardner, R. W., Garera, S., Pagano, M. W., Green, M., and Rubin, A. D. (2009). Securing Medical Records on Smart Phones. In *Proceedings of the First ACM Workshop on Security and Privacy in Medical and Homecare Systems*, SPIMACS '09, pages 31–40.

Hester, J., Peters, T., Yun, T., Peterson, R., Skinner, J., Golla, B., Storer, K., Hearndon, S., Freeman, K., Lord, S., Halter, R., Kotz, D., and Sorber, J. (2016). Amulet: An Energy-Efficient, Multi-Application Wearable Platform. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems*, SenSys '16, pages 216–229.

Jafari, M., Safavi-Naini, R., and Sheppard, N. P. (2011). A Rights Management Approach to Protection of Privacy in a Cloud of Electronic Health Records. In *Proceedings of the 11th Annual ACM Workshop on Digital Rights Management*, DRM '11, pages 23–30.

Knöll, M. and Moar, M. (2011). On the Importance of Locations in Therapeutic Serious Games: Review on current health games and how they make use of the urban landscape. In *Proceedings of the 2011 5th International Conference on Pervasive Computing Technologies for Healthcare and Workshops*, PervasiveHealth '11, pages 538–545.

Kouris, I. and Koutsouris, D. (2014). Identifying Risky Environments for COPD Patients Using Smartphones and Internet of Things Objects. *International Journal of Computational Intelligence Studies*, 3(1):1–17.

Kumar, S., Nilsen, W., Pavel, M., and Srivastava, M. (2013). Mobile Health: Revolutionizing Healthcare Through Transdisciplinary Research. *Computer*, 46(1):28–35.

Marcelino, L. and Silva, C. (2018). *Location Privacy Concerns in Mobile Applications*, pages 241–249. Springer, Cham.

Mare, S., Sorber, J., Shin, M., Cornelius, C., and Kotz, D. (2014). Hide-n-Sense: Preserving Privacy Efficiently in Wireless mHealth. *Mobile Networks and Applications*, 19(3):331–344.

Mattila, E., Korhonen, I., Salminen, J. H., Ahtinen, A., Koskinen, E., Särelä, A., Pärkkä, J., and Lappalainen, R. (2010). Empowering Citizens for Well-being and Chronic Disease Management with Wellness Diary. *IEEE Transactions on Information Technology in Biomedicine*, 14(2):456–463.

Milošević, M., Shrove, M. T., and Jovanov, E. (2011). Applications of Smartphones for Ubiquitous Health Monitoring and Wellbeing Management. *Journal of Information Technology and Applications*, 1(1):7–15.

Mishra, S. M. (2015). *Wearable Android: Android Wear and Google FIT App Development*. Wiley Online Library, Hoboken, New Jersey.

Moehrke, J. (2017a). Audit Trail and Node Authentication. Technical report, IHE International. https://wiki.ihe.net.

Moehrke, J. (2017b). Mobile access to Health Documents (MHD). Technical report, IHE International. https://wiki.ihe.net.

Murad, A., Schooley, B., and Abed, Y. (2013). A Secure mHealth Application for EMS: Design and Implementation. In *Proceedings of the 4th Conference on Wireless Health*, WH '13, pages 15:1–15:2.

Murnane, E. L., Huffaker, D., and Kossinets, G. (2015). Mobile Health Apps: Adoption, Adherence, and Abandonment. In *Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers*, UbiComp/ISWC '15 Adjunct, pages 261–264.

O'Donoghue, J. and Herbert, J. (2012). Data Management Within mHealth Environments: Patient Sensors, Mobile Devices, and Databases. *Journal of Data and Information Quality*, 4(1):5:1–5:20.

Schweitzer, J. and Synowiec, C. (2012). The Economics of eHealth and mHealth. *Journal of Health Communication*, 17(Supplement 1):73–81.

Siewiorek, D. (2012). Generation Smartphone. *IEEE Spectrum*, 49(9):54–58.

Silva, B. M., Rodrigues, J. J., de la Torre Díez, I., López-Coronado, M., and Saleem, K. (2015). Mobile-health: A Review of Current State in 2015. *Journal of Biomedical Informatics*, 56(C):265–272.

Stach, C. (2013). How to Assure Privacy on Android Phones and Devices? In *Proceedings of the 2013 IEEE 14th International Conference on Mobile Data Management*, MDM '13, pages 350–352.

Stach, C. (2015). How to Deal with Third Party Apps in a Privacy System — The PMP Gatekeeper. In *Proceedings of the 2015 IEEE 16th International Conference on Mobile Data Management*, MDM '15, pages 167–172.

Stach, C., Dürr, F., Mindermann, K., Palanisamy, S. M., Tariq, M. A., Mitschang, B., and Wagner, S. (2017). PATRON — Datenschutz in Datenstromverarbeitungssystemen. In *Informatik 2017: Digitale Kulturen, Tagungsband der 47. Jahrestagung der Gesellschaft für Informatik e.V. (GI), 25.9-29.9.2017, Chemnitz*, volume 275 of *LNI*, pages 1085–1096. *(in German)*.

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*, MDM '13, pages 305–313.

Stach, C. and Mitschang, B. (2014). Design and Implementation of the Privacy Management Platform. In *Proceedings of the 2014 IEEE 15th International Conference on Mobile Data Management*, MDM '14, pages 69–72.

Stach, C. and Mitschang, B. (2016). The Secure Data Container: An Approach to Harmonize Data Sharing with Information Security. In *Proceedings of the 2016 IEEE 17th International Conference on Mobile Data Management*, MDM '16, pages 292–297.

Steimle, F., Wieland, M., Mitschang, B., Wagner, S., and Leymann, F. (2017). Extended provisioning, security and analysis techniques for the ECHO health data management system. *Computing*, 99(2):183–201.

Stollmann Entwicklungs- und Vertriebs-GmbH (2014). Terminal I/O Profile: Client implementation guide. Technical report, Telit.

Weerasinghe, D., Rajarajan, M., and Rakocevic, V. (2009). *Device Data Protection in Mobile Healthcare Applications*, pages 82–89. Springer, Berlin, Heidelberg.