An Open Platform for Personal Health Record Apps with Platform-Level Privacy Protection

P. Van Gorp\textsuperscript{a,∗}, M. Comuzzi\textsuperscript{b}, A. Jahnen\textsuperscript{d}, U. Kaymak\textsuperscript{a}, B. Middleton\textsuperscript{c}

\textsuperscript{a}Eindhoven University of Technology, The Netherlands
\textsuperscript{b}City University London, United Kingdom
\textsuperscript{c}Partners HealthCare and Harvard Medical School, MA, USA
\textsuperscript{d}Public Research Center Henri Tudor, Luxembourg

Abstract

One of the main barriers to the adoption of Personal Health Records (PHR) systems is their closed nature. It has been argued in the literature that this barrier can be overcome by introducing an open market of substitutable PHR apps. The requirements introduced by such an open market on the underlying platform have also been derived. In this paper, we argue that MyPHRMachines, a cloud-based PHR platform recently developed by the authors, satisfies these requirements better than its alternatives. The MyPHRMachines platform leverages Virtual Machines as flexible and secure execution sandboxes for health apps. MyPHRMachines does not prevent pushing hospital- or patient-generated data to one of its instances, nor does it prevent patients from sharing data with their trusted caregivers. External software developers have minimal barriers to contribute innovative apps to the platform, since apps are only required to avoid pushing patient data outside a MyPHRMachines cloud. We demonstrate the potential of MyPHRMachines by presenting two externally contributed apps. Both apps provide functionality going beyond the state-of-the-art in their application domain, while they did not require any specific MyPHRMachines platform extension.

Keywords: Personal Health Records, Apps, Architecture, Trust, Privacy

1. Introduction

Without the participation of the patient, a health care provider cannot effectively treat (or prevent) disease-causing behaviors. The doctor-patient relationship is therefore gradually evolving from a paternalistic approach to a more participatory model [1, 2]. Houston and Ehrenberger argue that a key factor for successful patient participation is information sharing: patients require good information not only to care for themselves, but also to effectively communicate

∗Corresponding author

Email address: p.m.e.v.gorp@tue.nl (P. Van Gorp)
with their physicians [3]. Empowering the patient with information is particularly important since information exchange between different caregivers is very limited [4], especially beyond the scope of local business networks (such as the Partners HealthCare system in the US state of Massachusetts or the The Eye Care Network in the Netherlands [5, 6]).

The two key stakeholders in this scenario, i.e., patients and their physicians, are often willing and capable to share information. Already before the turn of the millennium, for instance, various online surveys demonstrated high adoption rates of e-mail as a patient-provider communication medium [7]. E-mail information sharing, unfortunately, has several limitations. Most notably, message exchanges are completely ad-hoc, preventing patients to build and maintain a longitudinal record of their health data, to use the integrated record to effectively care for themselves, and to share all their health data effectively and securely with their caregivers.

To overcome these limitations, Personal Health Record (PHR) systems have been proposed by various companies and authors in academia [8]. PHR have many societal benefits, such as empowering patients in the management of their own health and fostering interoperability among health care providers, possibly reducing the overall costs of diagnosis and treatment [9]. Policy makers, therefore, have repeatedly called for technologies that “enable patients, doctors and other health care providers to access personal health records securely through the Internet, no matter where a patient is seeking medical care” [10, 11]. Unfortunately, PHR adoption levels in practice are very low due to privacy concerns as well as the lack of convincing medical and business use cases. The US department of Health and Human Services, for instance, has invested heavily with the expectation that “once the market has structure, patients, providers, medical professionals and vendors will innovate, create efficiencies and improve care” [10].

One of the reasons for the low level of adoption of PHRs is their lack of openness at the platform level. Mandl et al. [12] have addressed the issue by looking at positive and negative experiences from various health record projects. The authors conclude that PHR technologies should go beyond the “conventional” requirements for Electronic Health Record (EHR) technologies, i.e., interoperability, security, and privacy. PHR systems should support open innovation and, therefore, they should (a) reduce impediments to the transfer of data, (b) provide substitutable software components, i.e. “apps”, and (c) they should allow competition and “natural selection” for high-value, low-cost software components. Regarding substitutability, the authors clarify that PHRs should enable the combination of software components developed by different vendors without creating impediments to replace such components over time [12].

In this paper, we propose the use of MyPHRMachines, a PHR platform that satisfies the above requirements. The platform is unique in its openness: it presents the least possible impediments to the transfer of data and it prevents apps from violating privacy requirements by design. These properties are based on the use of Virtual Machines (VMs) as flexible and secure execution sandboxes for the apps. To show the effectiveness of the approach, we discuss externally
contributed apps for Radiation Exposure Measure (REM). As we will show later, radiology and, more specifically, REM, is a typical application scenario that can benefit from an open PHR platform.

The remainder of this paper is structured as follows. Section 2 discusses the shortcomings of current PHR platforms with regards to openness. Section 3 describes the MyPHRMachines platform. Section 4 gives the motivation for and presents the REM application scenario, while Section 5 describes the REM apps in MyPHRMachines. Finally, Section 6 discusses the contribution of the paper by providing a link also to the PHR literature.

2. Openness of PHR Platforms

Opening a platform enables its owners to strategically disclose aspects related to the development or commercialization of the platform [13].

There are broadly two different approaches to opening a platform. The first entails giving up some control over the platform, whereas the second entails only granting access to the platform to outsiders [14]. When a company devolves all control over a platform, there is no longer a single party who controls its evolution. In terms of PHR platforms, this would mean for example that the development activities for a platform are opened up to the open source community, or to selected commercial software vendors. The Indivo platform is the primary example of this form of PHR platform openness [15]: starting from a development project at the Harvard Medical School and Massachusetts Institute of Technology, the project was then opened to the open source community as well as to Google, Microsoft and other commercial partners.

The second form of openness (granting access) implies that the platform owner maintains control over its core development while relying on the market to provide complementary innovation around it. Apple’s App store is a well known example of this approach, where the company not only preserves control over the platform’s development, but even controls the transactions on the platform. Microsoft HealthVault is a well known PHR platform that is open to apps from third party developers, while Microsoft controls the core platform [16].

We position the novelty of our PHR platform in the latter category. MyPHRMachines provides app developers with open access to the app platform, but it guarantees that patients can trust the platform in the protection of their personal data.

As illustrated in the remainder, other PHR platforms are either (1) completely closed or (2) pose too tight restrictions on the type of data that can be managed by the platform. In the latter case, technical guarantees regarding the prevention of data abuse are completely missing. Therefore, for those PHR platforms that grant app developers access to deploy their apps, access is only granted to trusted parties that can be held liable in case they violate their promises to the platform provider and end users. MyPHRMachines makes such app-specific trust considerations irrelevant, since technical privacy protection measures are already implemented at the platform level. Consequently,
a MyPHRMachines-based App store can be opened up securely also to non-trusted app developers.

PHR system architectures can be classified into provider-tethered and free-standing ones [17]. For the provider-tethered variant, the PHR system is essentially a portal extension of Hospital Information Systems (HISs), which only contain data from one health care provider or institution. Examples in this category are EPIC MyChart [18] and MyHealthCteVet [19], tethered from the EPIC EHR and the HIS of the US Department of Veterans Affairs, respectively. Free standing PHRs are stand-alone PHR platforms, which can store data generated and provided by various health care institutions or by the patient. Examples in this category are HealthVault and Indivo version X [15]. In principle, this classification only considers the stakeholder controlling the PHR platform (a single health organization versus an independent party). In practice, all tethered PHR systems are completely closed, while some free-standing PHR systems make their platform accessible to external app builders. Still, there are fundamental issues even for free-standing solutions. Below, we discuss some of these issues for the cases of Microsoft HealthVault and Indivo X.

Microsoft HealthVault provides a set of libraries (e.g. for Java and .NET developers) to Create, Read, Update, and Delete (CRUD) all types of data in the HealthVault system. The libraries are based on a Web service API. Similarly, Indivo X enables external software to perform CRUD operations on its health data through XML-based standard data models. Indivo X is also integrated with SMART [20], a more general solution to support the exchange of health data among health institutions. SMART provides an OWL-DL ontology to semantically annotate health data. Unfortunately, for both platforms, two of the requirements elicited in Section 1 are not satisfied:

1. existing platforms do not actively prevent apps from violating end-user privacy requirements, and
2. existing platforms pose impediments on the transfer of health data.

The first issue relates to Mandl et al.’s conventional requirements for EHR systems, while the second one relates to their extra requirements for openness.

The privacy issue is caused by the fact that neither HealthVault nor Indivo X apps are executed inside a controlled ecosystem. Instead, app code is executed on a third party infrastructure and, if users grant an app access to load PHR data, then that data can travel freely to the servers of the app providers. In terms of liability, the platform providers (Microsoft and others) push responsibilities to the app builders and the end-users. This implies that (i) all app builders need to provide a terms of use agreement that promises that patient data will not be abused and (ii) end-users need to review and consent such agreements for each and every app. While such agreements can protect end-users ex-post (e.g., legally) they do not physically prevent app providers to maliciously use the PHR data behind the scenes. Also, for app builders not interested in patient data, this need for app-specific data use agreements forms an undesirable barrier to entering the app market.
The second issue, i.e., impediments on the transfer of health data, is caused by the fact that data can only be stored on the HealthVault or Indivo X servers if it strictly conforms to the data formats that have been selected by the platform providers. This is a fundamental limitation since it prevents a market-based evolution of such formats. As a practical example of the impediment, we observed that it is impossible to store radiology images in the Indivo X platform and in the European deployment of HealthVault. A practical negative consequence of this is that, for example in the case of HealthVault, many third party apps store data outside the platform’s data repository. This is in conflict with the substitutability of the PHR apps requirement, since only the apps from specific third parties will be able to access the radiology data. Although over time platforms such as Indivo X and HealthVault may address such limitations, we argue that fundamentally, there will always be medically meaningful data for which a competitive app market moves ahead of platform-imposed standard data formats.

In the remainder, we explain how MyPHRMachines overcomes these issues.

3. MyPHRMachines as an Open and Trustable PHR App Platform

In this paper we focus on the aspects of MyPHRMachines that make it an open platform. Other details about MyPHRMachines can be found in previous work of the authors [21, 22, 23].

MyPHRMachines is a cloud-based PHR system. It gives patients convenient access to remotely running virtual machines (VMs), which give access to all their PHR data. We informally define apps as light-weight applications that provide very focused functionality (as opposed to monolithic information systems). In this context, VMs are the MyPHRMachines-specific “app” technology. VMs run as a service on a trusted and powerful hardware infrastructure and fulfill the role of app containers. MyPHRMachines apps can be accessed from regular computers and from tablets or mobile phones.

Section 3.1 summarizes the technical architecture of MyPHRMachines, while Section 3.2 discusses specifically the privacy protection as a service enabled by the design of MyPHRMachines. An example app, i.e. a radiology image viewer, is presented in Section 3.3. While that example app has been contributed by the MyPHRMachines platform developers, the two apps from Section 5 are provided by third parties. Section 3.4 briefly explains the process of deploying new apps to MyPHRMachines.

3.1. Technical Architecture

MyPHRMachines has a layered architecture and reuses various robust components such as a commercially available hypervisor and an open source data cloud with interfaces to commercial data clouds. The platform is extremely flexible with regards to PHR data formats and middleware and it makes apps available as a service via thin client technologies.
3.1 Technical Architecture

Figure 1 shows the architecture of MyPHRMachines. At the highest aggregation level, MyPHRMachines comprises a client and a server layer. The server layer is further decomposed into an execution layer and a storage layer.

MyPHRMachines relies on thin clients: a client should be able to (1) access the app store, (2) run a viewer to work with remote VMs and (3) upload and download PHR data via the data cloud components. The three arrows leaving the Browser component in Figure 1 show that any device with HTML and Java support already satisfies the above three requirements, without any MyPHRMachines-specific software installation. The Native RDP Client represents native client software required to support function (2) by non-Java enabled clients, e.g., clients running iOS, which does not support Java at the time of writing. MyPHRMachines, in fact, relies on the standard Remote Desktop Protocol (RDP) to view remote VM sessions. The Native Own Cloud Client and Dropbox clients support function (3). For its data cloud functionality, MyPHRMachines relies upon off-the-shelf software from the mature OwnCloud project [24]. Besides providing secure storage within the MyPHRMachines infrastructure, the OwnCloud component also enables users to plug in their Dropbox or Google Drive for mounting less sensitive data [25] (see the link connecting elements of Dropbox to the Private Data Cloud).

In the Storage layer, Figure 1 shows also the Private Network Folders. While the private data cloud requires an ad-hoc virtual network to the OwnCloud server that runs within the MyPHRMachines infrastructure, these mounted folders are accessible even to VMs that do not have a network interface. Although the OwnCloud server is residing in a Demilitarized Zone (DMZ [26]), protected by a firewall, it is not as secure as the mounted folders mechanism since, in theory, an app could hack its way from the ad-hoc network to the OwnCloud.
server and then to the public internet. This is impossible for VMs without a network interface.

The execution layer of the MyPHRMachines server contains two components: the Web Portal (or AppStore) and the Hypervisor. The web portal manages the access control of users to apps. This can be therefore seen as the “app store” of MyPHRMachines. The app store also communicates with the Hypervisor, which is a generic piece of software to start, stop, and clone VMs and control their Internet access. MyPHRMachines currently uses VirtualBox, i.e. an off-the-shelf hypervisor, which is heavily used also in other industries, e.g., banking. Finally, messages between the app store and the hypervisor are delivered via SSH, a secure and stable communication protocol.

Patients can communicate with their health providers in two ways: first, they can delegate the remote access to a VM and, secondly, they can share the raw PHR data using the underlying data cloud. This private data cloud provides file sharing features similar to DropBox and Google Drive, but it also ensures that the physical location of the data is by default within the trusted MyPHRMachines infrastructure. This feature was added to MyPHRMachines recently (i.e., after the publication of our previous work [21, 22, 23]). It relates in general to removing impediments to the transfer of data between trusted parties (cf., the criteria from Section 1). In particular, it enables patients to grant their GP a copy of specific PHR files for the sake of accountability. Also, thanks to the OwnCloud sync client, patients can conveniently upload new PHR content (e.g., a copy of a new radiology CD).

Open innovation is supported by MyPHRMachines by the fact that any health care institution or software provider can contribute a new app by remotely cloning an existing VM image, installing the new software remotely, and publishing it to the app store. App developers can choose between accessing health data files directly from the VM file system or accessing data through a more heavy-weight Application Programmer Interface (API). The platform is flexible because any kind of middleware that runs on an operating system that can be virtualized can also be used to build apps.

VM sessions in MyPHRMachines are stateless, meaning that data that is written to the local disk of a VM will be deleted after VM shutdown. This enables app developers to update their VMs without worrying about migrating patient-specific VM sessions. MyPHRMachines does enable apps to create or update data persistently in a patient’s PHR. This is realized by means of a writable mounted folder in the patient’s VMs. Additionally, data can be persisted via the private data cloud.

3.2 Privacy Protection as a Platform Service

MyPHRMachines protects privacy as a platform service. Although privacy is seen as essential in the context of the requirements discussed in Section 1, no other platform provides technical mechanisms for this. This subsection clarifies that, in contrast to the novelty of having this service, its implementation is relatively simple to realize, given the architecture described in the previous subsection.
3.3 Example App: Radiology Image Viewer

MyPHRMachines can provide a privacy protection service thanks to its very design, which is based on the principle that software should be moved to data rather than vice versa. Once all software is available in the MyPHRMachines private cloud, apps no longer need access to external Internet services. The MyPHRMachines execution layer therefore enforces that published VMs have no network interface with Internet access. Even if a malicious app developer, for example, installs malware in a VM, such malware will fail to push data outside of the app container.

PHR platforms lacking the ability to completely block Internet access by apps have to rely on complex analyses of the data streaming out of their ecosystem (e.g., has the patient approved access to the data by the app builder? Is the app builder’s server properly authenticated? Is traffic properly encrypted? Can the data pass over servers that are subject to the US patriot act? Etc.).

Note that MyPHRMachines does not guarantee privacy protection in general. In particular, since MyPHRMachines aims to reduce impediments to the transfer of data, patients can choose to use the OwnCloud component in combination with a US-based cloud storage provider (e.g., DropBox), and therefore be subject, for instance, to the NSA scrutiny. Yet, MyPHRMachines does guarantee that PHR data is protected from app builders (and their governments). This implies that when a MyPHRMachines cloud is deployed on EU infrastructure, the NSA could not force US-based app builders to give access to PHR data on which their apps are applied. This example implication is of high political relevance in current times.¹

3.3. Example App: Radiology Image Viewer

Radiology tests are often repeated due to the loss of a test result or due to inconvenient provider access to the images. Besides being inconvenient and unhealthy for patients, this also represents a waste of insurance and taxpayers money. In order to avoid this waste, insurance companies can simply provide their patients free use of a specialized Microsoft Windows app (VM) in MyPHRMachines [21]. As illustrated below, that is sufficient for giving any specialist convenient online access to a patient’s radiology images.

Ge et al. recently presented a novel portal prototype to store and share radiology images under patient ownership [27]. The MyPHRMachines radiology image viewer app presented here provides the same functionality as that prototype with a simpler implementation of an app that is substitutable. The implementation of the app is simpler since it reuses viewer software that is already embedded in the patient’s radiology CD. Moreover, the functionality to give a physician access to the viewer is implemented at the MyPHRMachines platform layer. The app is substitutable since anyone can install a more advanced viewer to a new VM and offer that to other MyPHRMachines users.

Figure 2 shows tablet and laptop access to the image viewer app in MyPHRMachines. As explained in Section 3.1, the laptop provides zero-install access to

¹See http://ec.europa.eu/justice/data-protection/
the specialized app (since the app viewer relies on HTML and Java only). The
tablet is an iPad (for which Java support is not available). Hence, it relies
on a native RDP client for working with the remote VM. Multiple such RDP
clients are available regardless of MyPHRMachines and our radiology viewer
app. Therefore, also at this level, MyPHRMachines supports substitutability.
From the usability point of view, the use of a native RDP client does require
users to enter the address and port on which the remote VM is running. Android
tablets (which do support Java) do not have this potential usability barrier.

The delegation of access to radiology images does not require any app-specific
implementation since it is supported by a generic MyPHRMachines platform
feature, i.e. the app store portal (see Figure 1). That portal enables patients to
delgate access to any of their active VMs. More specifically, for every active
VM, patients can generate an automatic e-mail message which enables the re-
ceipient to log in to the remote VM without signing up for a MyPHRMachines
account [21].

Readers are encouraged to visit the companion Web site https://sites.
google.com/site/myphrmachines/ for hands-on access to this demo. The
demo provides anonymous access to a dummy account for which multiple radi-
ology CDs have been uploaded to the PHR.

3.4 Contributing a new App to MyPHRMachines

The companion Web site of this paper will be maintained to provide up-to-
date instructions for contributing new apps. In this paper, we abstract from the
rather volatile user interface details and focus on the conceptual workflow for
deploying new apps to MyPHRMachines. Technical details have been published before [22, 23], so we omit them here.

Figure 3 sketches the key activities related to new app provisioning. The diagram uses BPMN, an industrial standard notation for modeling business processes [28]. The upper swimlane shows tasks that are executed by representatives of an external software vendor while the lower one displays tasks that are executed by employees of an organization offering the MyPHRMachines platform services. The current deployment of MyPHRMachines is maintained for academic demonstration purposes only. However, anybody can contact the authors for leveraging that demonstrator infrastructure.

The workflow model shows at its top left an empty circle with thin edge, representing the process start event. The first task in the process (i.e., “Request VM Clone”) is executed by the app builder. As explained in Section 3.1, the web portal enables such app builders to request a clone of an existing virtual machine. The two external apps presented in Sections 5.2 and 5 are based on clones of Windows and Linux virtual machines respectively.

MyPHRMachines VMs are organized in groups. This is important since otherwise all apps would be visible in one global namespace, which would not scale. Each group has at least one administrator. The “Group Admin” lane in Figure 3 models the tasks that should be executed by such administrators, in the context of the new app deployment workflow. The dashed arcs between different lanes represent messages that are sent by the MyPHRMachines portal. The workflow can only proceed from one task A to a successor task B if (1) A is completed and (2) for each incoming message flow in B, a message has been received. Following this semantics, Figure 3 sketches that the app builder can only deploy his binaries to the cloned VM after the clone request was approved by an administrator, and after that administrator has moved the cloned VM to
a private group. The purpose of such group is to contain virtual machines that are not yet ready to be published in the app store.

Figure 3 includes three tasks that are labeled “Test VM”. The tasks are in the lanes of the app builder, an alpha tester and a beta tester respectively. All testers use exactly the same software, i.e., exactly the same VM configuration, but their VM instances will be initialized with their own individual test data. Each tester can upload test data using the same functionality that end-users use to upload PHR data. If either the alpha tester or app builder think the VM is not ready yet for a release to the app store, the workflow moves back to task “Deploy Binaries”. Upon each entry of that task, the app builder gets private and mutable access to the VM configuration. When completing the task, the VM configuration is saved such that each tester and subsequent user will start from the same software configuration context.

When the app builder decides to publish a VM, a group administrator moves it to a public group. Finally, the VM is made available in the app store. Optionally, a quality check can first be performed by platform maintenance staff. While the workflow in Figure 3 focuses on the general case, MyPHRMachines manages also the the access rights and stakeholder notifications for the various tasks.

4. Radiation Exposure Monitoring

This section introduces our application scenario of Radiation Exposure Monitoring (REM). We first discuss the need for REM in Section 4.1 and then provide more technical details about the measurement of radiation exposure in Section 4.2.

4.1. The Need for Patient-Level REM Services

X-rays have been officially classified as a carcinogen by research agencies and prevention centers [29, 30]. The presumption is that significant increase in the population’s cumulative exposure to ionizing radiation will cause an increased incidence of cancer years down the line. To prevent this, it is not advisable to take a passive data collection and prevention approach, since “Radiation-induced cancers typically do not occur until 1 or 2 decades or longer after exposure.” [31] The largest epidemiologic study currently available shows a statistically significant increase in cancer at radiation dose estimates in excess of 50 mSv [32]. Many computed tomographic (CT) scans and nuclear medicine studies have effective dose estimates whose cumulative doses easily exceed this level [31]. Related knowledge is gradually expanding, among others via international cohort studies [33]. In the meanwhile, “the current annual collective dose estimate from medical exposure in the United States has been calculated as roughly equivalent to the total worldwide collective dose generated by the nuclear catastrophe at Chernobyl” [31, 34, 35].

REM aims at monitoring the level of exposure to ionizing radiation by patients. Over the last decades, significant progress has been achieved by monitoring mean exposure values against so-called Diagnostic Reference Levels (DRLs).
Rehani for example shows that initially some countries had very disturbing high exposures whereas results were more harmonized after alerting these issues and acting upon them [36]. This is all however at the national policy level and the exposures for individual patients are not yet properly governed. The current absence of patient-level monitoring mechanisms means that individual patients that receive dangerously high exposures will remain un-noticed by today’s radiology information systems. To the best of our knowledge, this problem has not received much research attention yet, but one can easily conceive high cancer risk scenarios.

From a healthcare informatics point of view, it is indeed challenging that patients are likely to undergo scans at different health care institutions during their life. In contrast, organizing REM support around the patient is very natural since patients can track exactly the number and type of scans they have undergone. We argue that patients should therefore be empowered with PHR-based REM application software, enabling them to monitor their radiation exposure over time and discuss it with their caregivers. This patient empowerment enables doctors to make better risk assessments and specialize the diagnosis and treatment plan.

Large radiology technology vendors are aware of the aforementioned issues. However, Brosky has recently clarified on a professional radiologist community website why it is unrealistic that they will provide integrated REM services soon [37]. The author describes the results of a European vendor-focused integration workshop. Although various vendors are offering standard-compliant dose reporting products, these products are deemed to fail on today’s market. On the one hand, there is a vast installation base of legacy products that (1) do not support the standard reporting interfaces and (2) are not expected to be phased out. Moreover, even when a standard-compliant reporting system is used, healthcare administrators still need to configure (or program) the data transfers between local and centralized REM systems. At such integration workshop, only one of eight participating companies provided “a full-scale dose information reporter that enables a healthcare system to look into the accumulated data and extract actionable information.” [37]. Other vendors indicated that “If we build it now, no one will buy it”.

4.2 Calculating the Cumulative Estimated Dose of Radiology Absorption

There is a key difference between the radiation exposure from the various imaging modalities and the actual amount of radiation absorbed by a patient. The latter is dependent on the amount and properties of each tissue encountered by the X-ray beam. As it is not practical to insert radiation detectors into each organ of every patient, absorbed radiation dose is measured only directly in extreme cases of oncology treatment. Therefore, there is a great need to support the accurate estimation of absorbed radiation doses.

Promising results have been achieved in the area of automatic CED calculation. More specifically, recent software programs enable the automatic extraction and analysis of dose-related parameters from image meta-data. So far, these programs have only been used within complex pipelines. For example,
Jahnen et al. [38] use such an extraction component within the PerMoS chain, which is primarily designed to monitor DRL conformance at the governmental level. Aware, inc, provides a chain that also relies on a central dose index registry. The Aware chain does include components at finer granularity levels: it provides support for monitoring the conformance of individual technician and physicians and also aims at risk management for individual patients [39]. However, for monitoring patient-level exposures, the Aware chain assumes that all hospitals visited by the patient push their data to the central registry. As clarified by Brosky [37], this is unfortunately not realistic.

At the core of the aforementioned chains are, however, extraction and analysis components that would provide patient-level CED calculations when provided with patient-level data. In this paper, we use MyPHRMachines to provide patient-level data securely to the extraction and analysis components of PerMoS (Tudor) and Aware.

5. REM Apps in MyPHRMachines

We have deployed the two alternative CED management components of the Tudor and Aware chains apps in MyPHRMachines. Both apps visualize received dose values in a patient-centered manner. In the following, we first discuss the input format requirement of both apps. We then discuss the key functionalities of the individual apps. Finally, we reason about the implications of having these two demonstrators.

5.1. App Input: DICOM

The DICOM standard defines a file format for storing radiology data on physical media (e.g., CD ROMs) as well as a communication protocol for transferring images from/to remote servers.

**File Format** DICOM prescribes a standard format for storing radiology images. Additionally, the standard prescribes how to store information about the image data. Such information is called “metadata”. Besides standardizing metadata for characterizing the patient (e.g., name), the standard also prescribes metadata fields for storing technical equipment and machine parameters (e.g., scanner model, scan length, scan modality and scan location). When using full digital equipment, dose information is stored explicitly in DICOM fields too. For older equipment types however, such information is missing and therefore the aforementioned simulation methods need to be employed. The first app (MyPHRDoseReporter) supports full digital equipment as well as legacy equipment while the second app only supports full digital equipment.

**Communication Protocol** The second app includes a VM startup script that automatically collects all the user’s radiology data and sends that via standard DICOM protocol messages to the Aware REM server that is running locally in the remote VM.
5.2 Tudor App: MyPHRDoseReporter

MyPHRDoseReporter is an application supporting the visualization and management of medical images. The application integrates various open source libraries developed by the Public Research Centre “Henri Tudor” into a patient-centered app. The app supports both the construction of a personal radiology record as well as the inspection thereof. Regarding record building, the app can import data from (virtualized) patient CDs. The app can harmonize input data in order to overcome differences in the implementation of the DICOM standard by different scanner manufacturers. Regarding inspection, the app supports both the interactive viewing of radiology images as well as the calculation of estimated dose values for various modalities and machine brands. Figure 4 shows a screenshot with cumulative dose results grouped by imaging modality.

![DICOM Viewer](image1.png)

From an end-user’s perspective, the second aspect is an implementation detail. What does matter for end-users is that both apps require input data stored in the DICOM file format.

5.2. Tudor App: MyPHRDoseReporter

MyPHRDoseReporter is an application supporting the visualization and management of medical images. The application integrates various open source libraries developed by the Public Research Centre “Henri Tudor” into a patient-centered app. The app supports both the construction of a personal radiology record as well as the inspection thereof. Regarding record building, the app can import data from (virtualized) patient CDs. The app can harmonize input data in order to overcome differences in the implementation of the DICOM standard by different scanner manufacturers. Regarding inspection, the app supports both the interactive viewing of radiology images as well as the calculation of estimated dose values for various modalities and machine brands. Figure 4 shows a screenshot with cumulative dose results grouped by imaging modality.

The app can manage input from various radiology imaging modalities, as illustrated by Table 1. Based on an inspection of all images in a patient record, the app generates a tabular overview of the received dose. The Computed Tomography (CT) modality involves rotating beams, while the others involve uni-
5.2 Tudor App: MyPHRDoseReporter

<table>
<thead>
<tr>
<th>Modality</th>
<th>DICOM Identification</th>
<th>Dose Figure</th>
<th>Meaning of Dose Figure</th>
<th>Extracted (X) or Calculated via Simulation (S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computed Tomography</td>
<td>CT</td>
<td>CTDIvol</td>
<td>Computed Tomography Dose Index, Volumetric</td>
<td>X if available, S otherwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DLP</td>
<td>Dose Length Product</td>
<td>X if available, S otherwise</td>
</tr>
<tr>
<td>Digital Radiology</td>
<td>DR</td>
<td>DAP</td>
<td>Dose Area Product</td>
<td>X</td>
</tr>
<tr>
<td>Fluoroscopy</td>
<td>DF</td>
<td>DAP</td>
<td>Dose Area Product</td>
<td>X</td>
</tr>
<tr>
<td>Angiography</td>
<td>XA</td>
<td>DAP</td>
<td>Dose Area Product</td>
<td>X</td>
</tr>
<tr>
<td>Mammography</td>
<td>MR</td>
<td>MGD</td>
<td>Mean Glandular Dose</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 1: Dose figures for the different modalities in MyPHRDoseReporter.

directional beams. The *Computed Tomography Dose Index volumetric* (CTDI-vol) value is used as a dose descriptor per CT volume. One CT scan consists of multiple such volumes and each volume can differ in beam intensity. The *Dose Length Product* (DLP) is used to quantify the complete dose for a CT scan. In CT, doses are taken directly from the DICOM metadata (if available) or from a Monte Carlo simulation-based application otherwise (i.e., CT Expo [40]). For Diagnostic Radiology (DR), Fluoroscopy (DF) and Angiography (XA), MyPHRDoseReporter computes the *Dose Area Product* (DAP), which describes the dose quantity per square centimeter. For Mammography (MG), MyPHRDoseReporter extracts the *Mean Glandular Dose* (MGD), i.e. the mean dose to the glandular tissue of the scanned breast. As most MG machines are full digital, no Monte Carlo based simulation methods are implemented for this modality.

All metrics from Table 1 are well known to radiology specialists. Moreover, since specialists tend to use their own reference values for these metrics, depending on hospital protocols and national guidelines, MyPHRDoseReporter simply presents the raw metric results and leaves the interpretation of the data to the app user. In the long term, the following issues need to be tackled:

1. understanding which dosimetry concepts are of relevance to patients,
2. understanding which dosimetry concepts are useful for radiology nurses,
3. understanding how the degree of uncertainty can be properly presented.

These issues are the subject of ongoing research at Tudor. The app functionality can be extended by (i) including appropriate reference values in the output report, (ii) aggregating Effective Dose values, (iii) comparing the received individual dose to easy understandable facts, (iv) adding support for additional modalities, such as Nuclear Medicine activities, and (v) taking into account the current age of the patient.
5.3 Aware App: MyAccuradREMServer

The Aware Accurad REM Server is a software suite that has been designed for empowering radiologists with REM support. The server is typically connected to all radiology equipment of a hospital and possibly to the servers of other hospitals. One server typically contains the data of all patients that are known to the hospital. The MyAccuradREMServer app is a patient-centered VM deployment of such a server. The patient can access all calculated dose figures or delegate access to a specialist.

In contrast to MyPHRDoseReporter, MyAccuradREMServer does not provide simulation-based estimations based on the DICOM data from legacy scanners. However, it provides a more convenient user interface. Besides providing convenient table and graph filtering widgets, the Aware app provides workflows for defining and monitoring radiology protocols.

Figure 5 shows the app’s visualization of various cumulative dose results for a dummy patient. The figure shows that the cumulative dose can be displayed per target region (head versus lumbar spine, in the example).

5.4 Evaluation

From an evaluation perspective, we stress that (i) both apps have been developed and deployed by stakeholders external to the MyPHRMachines project and (ii) the two apps do not require any extension to the MyPHRMachines platform to function. This illustrates that the platform is indeed open to external functionality. We also stress that the platform is unaware of the format of the data consumed by the apps (i.e., DICOM content). This is in contrast to the requirements that other app-oriented PHR platforms (e.g., HealthVault and Indivo X) impose on input and output data formats. Beyond the PHR context, both apps demonstrate that an open, patient-oriented approach to Health Informatics may empower caregivers with functionality that may not available in HISs. Using the example apps discussed in this paper, patients can indeed present cumulative dose reports to their radiologist. Such reports cannot usually be generated by the HIS used by the specialist. Even hospitals equipped with a state-of-the-art REM server typically are still lacking integration with servers of other hospitals (cf., Section 4.2).

6. Discussion

This section discusses how this paper relates to the PHR literature. In particular, Section 6.1 discusses the advantages and potential pitfalls of using VMs as a more general PHR platform technology. Section 6.2 focuses on literature about PHR adoption barriers and facilitators. We include that section to clarify how we have leveraged adoption studies from other PHR systems in the design of MyPHRMachines.
Figure 5: Aware’s MyAccuradREMServer app in MyPHRMachines.
6.1 Strengths and Potential Pitfalls of Using Virtual Machines as Apps

This section discusses the key strengths of the VM-based architecture as well as pitfalls that should be avoided by app developers. The key strengths of the architecture are its flexibility and trustability. The flexibility strength is illustrated by the DICOM viewer example: zero-programming effort was required to deploy an off-the-shelf DICOM viewer into MyPHRMachines. This flexibility is in strong contrast to the PHR architectures with restrictive APIs (such as Indivo X [15]), which would require a significantly higher effort to build and maintain the apps described in this paper. The trustability strength follows from the aforementioned platform feature, which makes it impossible for apps to send patient data to external servers.

The flexibility strength, however, comes with a pitfall. Since the MyPHRMachines platform does not impose the use of standard data formats, a naive use of the platform would result in patient records riddled with syntactically or semantically incompatible fragments. App developers aware of this pitfall, however, may turn it into an enabler, by deploying apps to translate health record fragments into a proprietary format (or in the format of a deprecated standard) or into the latest standard format. Again, we argue that the ability to obtain such functionality from the App market makes the architecture more scalable than architectures where only the platform owners can provide data conversion functionality. Note that the fact that MyPHRMachines does not impose any data standards does not prevent the use of standards. In this paper, for example, we leverage DICOM as a standard for storing radiology data.

Another pitfall along a similar direction is that MyPHRMachines app developers may re-build low-level functionality, rather than reuse middleware features provided by platforms with heavyweight APIs. We argue that app developers should install inside their VM any meaningful middleware they can afford. For example, developers of diabetes-specific apps may want to deploy SMART middleware to their virtual machine [20]. This would provide them with libraries to manage lab data coded using the Logical Observation Identifiers Names and Codes (LOINC) standard.

The MyPHRMachines trustability strength also comes at a cost. By blocking general Internet access for patient-instantiated VMs, apps cannot by default leverage public Web services. This is unavoidable if one aims at fully dependable, platform-provided privacy governance of patient data. The example apps described in this paper do not require such services. For the sake of generalizability, we briefly discuss here how MyPHRMachines supports the use of public Web services in apps.

MyPHRMachines enables providers of stateless Web services to deploy their service in long-running VMs inside the MyPHRMachines cloud. Such VMs can be made available securely to patient-specific VMs. This is the same design choice we adopted for the OwnCloud service discussed in Section 3.1, which also runs in a long-running VM. In some practical cases, providers of very popular Web services could refuse to offer their service in this way. If the providers of such services are considered trustworthy, then VMs can be given controlled
Internet access to a specific domain. Careless use of this platform feature is however a pitfall that may decrease the trustability of MyPHRMachines apps.

6.2 MyPHRMachines and PHR adoption barriers

In relation to the issues described in Section 2, McGraw acknowledges that today’s PHR systems too often impose consent contracts that jeopardize patient privacy [41]. Kahn et al. also highlight that providers of PHR software services are not necessarily subject to US privacy laws and therefore are not as trustworthy as conventional care providers [42]. Another study by Witry et al. also reveals concerns about the consent contracts imposed by today’s health record systems [43].

MyPHRMachines provides privacy at the platform level, making “app”-specific privacy contracts irrelevant. According to McGraw, that should bolster trust in such systems and promote their adoption [41]. Patel’s survey [44] investigates the types of privacy threats patients are really concerned about. In that study, 94% of the surveyed patients had no privacy concerns towards their physicians [44], while respondents did have significant privacy concerns towards insurers, employers and the (US) government. To the best of our knowledge, no scientific studies have been conducted to analyze whether Patel’s results are valid beyond the US context. Therefore, it remains unclear which company or organization provides the right trust level for providing the MyPHRMachines platform professionally at a European or global scale. However, if within a specific context one trusted party offers the MyPHRMachines platform, then all the untrusted parties will be able to effectively deploy apps to that ecosystem. In particular, if patients do not trust the app providers, the MyPHRMachines platform guarantees that apps will not send data to the untrusted parties.

Kahn et al. conclude that the key technical adoption barrier for PHRs is that patients have to provide a large amount of information manually, using tedious and error-prone Web forms [42]. Other studies report that doctors are concerned about patients being too poorly informed to understand the meaning of their PHR data [45, 46]. Family physicians interviewed by Witry et al. therefore viewed potential in PHRs as a backup source of medical information secondary to the patient’s medical record as opposed to a tool for patient self-care [43]. Also, a physician from that study expressed that “For quality and efficiency it is worth it” and “It is worth it to give it to people for free. You’d save the (US) government money.”. In the context of this paper, we envision that in the long term, data could be pushed automatically to the PHR once produced in a hospital (or once a copy arrives at a patient’s GP). In our radiology use case, we currently expect patients to upload the content of a radiology CD. Patients have a good incentive to upload the data, as they do not have to worry anymore about preserving the physical CD. The MyPHRMachines platform also provides technical interfaces (beyond the scope of Section 3.1) enabling hospitals to send the radiology data directly, saving them the time and costs of creating CDs and sending them to patient homes.

Another study concludes that family physicians are quite open to sharing information with patients, as long as the related information systems are easy
to use and as long as their value to the practice of medicine has been demonstrated [46]. Regarding usability, we have also taken into account study results by Witry et al. [43]. The authors conclude that physicians are concerned about the time it takes to log into a PHR system and lookup specific information. We have taken this concern into account by enabling patients to (1) collect specific information as preparation for a time-critical doctor meeting and (2) provide one-click physician access to that specific information, as explained in Section 3.3. Regarding medical relevance, we stress that the REM apps provide unprecedented support for personalizing patient safety in radiology (cf., Section 4.1, which stresses the importance of REM in the context of cancer prevention).

7. Conclusions

In this paper, we demonstrated the potential of MyPHRMachines in terms of its openness. The MyPHRMachines platform satisfies requirements that had been identified previously by Mandl et al. and that existing platforms do not currently implement. In particular, PHR platforms suffer from weaknesses related to the types of data supported as well as to the way in which they handle data privacy.

MyPHRMachines leverages VMs as flexible and secure execution sandboxes. We have demonstrated the flexibility of the platform by showing that external developers could deploy apps that deal with data that cannot be handled by the repositories of other PHR platforms. The privacy strength follows from the platform design. The VM sandboxes reside in a private cloud and apps are not allowed to push data outside their sandbox. In our future work, we will evaluate MyPHRMachines in controlled clinical settings. We will also refine our example apps and new demonstrators will be designed and implemented, using data from various types of devices. Finally, we will investigate which organization has sufficient user level trust to host the MyPHRMachines platform.

Acknowledgements

The authors wish to thank Joe Kushi and James Cialdea from Aware, Inc. for contributing the MyAccuradREMServer app. Moreover, thanks to Adrian Gropper from HealthURL for the valuable discussions on cloud technologies.

References


21


