ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

On the design and analysis of protocols for Personal Health Record storage on Personal Data Server devices

Kirill Belyaev, Wuliang Sun, Indrakshi Ray *, Indrajit Ray

Department of Computer Science, Colorado State University, Fort Collins, USA

HIGHLIGHTS

- The architecture of Personal Data Server overlay is presented.
- The data is replicated in user-controlled secure portable tokens.
- Properties of the protocol are demonstrated using UML and Alloy.

ARTICLE INFO

Article history:
Received 16 September 2015
Received in revised form
19 May 2016
Accepted 22 May 2016
Available online xxxx

Keywords:
PHR
PDS
Alloy
UML

ABSTRACT

The electronic Personal Health Records (PHRs) such as medical history, lab reports, and insurance are stored in systems such as Microsoft Health Vault where a medical care provider or a patient is responsible for uploading and managing the health information. Storing PHRs in such a manner prohibits the patients from having complete control over their data and also may make the PHR system the target of security attacks. Towards this end, we proposed a new architecture, namely Personal Data Server overlay, where the data is stored on a set of Secure Portable Tokens (SPTs) that are under the control of individual users. SPTs are cheap, portable, and secure devices that combine the computing power and tamper-resistant properties of the smart cards and the storage capacity of NAND flash memory chips and they can act as a Personal Data Server (PDS).

We need formal assurance of data availability when information is stored in PDS overlays. Thus, data must be replicated at multiple PDSs. We propose a data replication protocol that ensures that the PHRs for each user have replicas in the PDS overlay. It is crucial to ensure correctness of the data replication protocol. Consequently, we formalize the protocol using the Unified Modeling Language (UML) and specify a number of desirable properties. We need to provide formal assurance of these properties in an automated manner. We demonstrate how the UML model can be transformed into Alloy using the UML-to-Alloy transformations. This obviates the need for the protocol designer to know Alloy. The analysis uncovers a significant error in the protocol. Uncovering such errors help refine the protocol and ensures its correctness before deployment.

1. Introduction

In the digital age, there is a plethora of personal information that individuals need to store and access. Consider the storage of Personal Health Records (PHR) as a motivating example. A PHR system can store diverse information related to public health, such as, medical history and lab reports, insurance, consent forms, and other relevant information. The use of PHRs can improve the health care of individuals and communities in both developed and developing countries [1] and there are over 200 different PHR systems in the USA [2].

We have various types of PHR systems, some of which are cloud based such as Microsoft Healthvault [3], where the user is responsible for uploading and managing her health information. Storing sensitive personal data, such as health data, on current cloud systems, such as Microsoft Healthvault [3], improves upon the systems supported by providers in that the users are responsible for managing their own data. However, such a solution, has several problems, including the potential for security and privacy breaches, that we outline below. First, service downtime and unavailability of Internet connection may limit an individual from accessing her data stored on the cloud. Clearly, this may

* Corresponding author.
E-mail addresses: kirill@cs.colostate.edu (K. Belyaev), sunwl@cs.colostate.edu (W. Sun), iray@cs.colostate.edu (I. Ray), indrajit@cs.colostate.edu (I. Ray).

http://dx.doi.org/10.1016/j.future.2016.05.027
0167-739X/

Please cite this article in press as: K. Belyaev, et al., On the design and analysis of protocols for Personal Health Record storage on Personal Data Server devices, Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.05.027

be unacceptable to many users who expect their data to be available all the time. Second, such servers storing PHRs of a large number of users are also target for attackers who have much to gain by compromising the privacy of health data. Note that, attackers can be insiders as well, such as dishonest or disgruntled employees of the organization. Third, the individuals must trust these organizations for adequate protection of the data against security and privacy breaches. Typically, users are reluctant to place trust on these organizations [4]. Fourth, even though an individual trusts an organization and is willing to comply with its security and privacy policies, such policies are subject to change with time. Moreover, security and privacy policies will almost certainly be changed if there is an acquisition or merger of companies. It is not clear how such a change will impact the existing records of an individual user and whether such a change will respect the original preferences of the user. Fifth, the servers storing data of a very large number of users must be kept online, which requires enormous power consumption. Consequently, we propose an alternative approach for storing PHR data.

Our approach builds upon the use of a new emerging technology of cheap ($10–20 range), portable (carry it in your pocket/purse) and secure devices that combine the computing power and strong, tamper-resistant security of smart cards and the ever increasing storage capacity of NAND flash memory chips. We call such devices Secure Portable Tokens (or SPTs in short). Examples of such SPTs can be found in wireless secure dongles, and smart USB jump drives with embedded chips. A very similar technology can be found in the SIM cards of mobile cellphones and smart phones.

Individuals own SPTs and store their personal data on the SPT, forming what we refer to as a *Personal Data Server* (PDS). The PDS alternative to existing PHR systems exploits the computing power of 32 bit RISC processors and strong, tamper-resistant security of smart cards.

1.1. Our contributions

Individuals have their PHRs on their own PDSs. For reasons of availability, the individuals may wish to replicate their PHRs on devices belonging to other individuals. The users have total control over what data they wish to replicate, where they would like to replicate, and may also change these decisions at any point of time. Thus, a user may replicate his data to one or more PDSs belonging to other trustworthy individuals. Each PDS, under the control of an individual user, may be online or offline depending on the user's discretion. To facilitate communication between two PDSs, we employ the services of an online *broker*. The broker simply provides a temporary storage service to which a PDS can upload data (publish) and from which another PDS can download data (subscribe). Replica propagation, in light of the fact that PDSs may be offline, poses a challenge. We have devised replica control protocols to address this issue. *Eventual consistency property* is assured provided the PDS holding the replica comes online after the update operation.

Queries may be posed on the PHR by individuals or organizations. Query processing poses similar challenges because a PDS containing the required PHR may be offline. In such a case, we demonstrate how the query response can be constructed from the PDS holding the original PHR or from the other PDSs containing the replica PHR. We try to minimize the response time for query processing, by trying to answer the query using the original or replica PDSs, whichever comes online first. We also guarantee the *data freshness* property that ensures that the response to the query is always accurate, even if some replicas may have not been online all the time.

We need to provide assurance that the system's behavior is as intended. Towards this end, we focus on the replication protocol because it forms the basis of data availability and reliability. We formalize the protocol using the Unified Modeling Language (UML) [5] notation and the Object Constraint Language (OCL) [6]. Our experience suggests that protocols that constrain how system elements are replicated can be conveniently expressed in class models augmented with OCL invariants and operation specifications. Structural aspects of these protocols can be expressed in terms of classes and relationships among classes, while functionality can be represented by operations associated with OCL pre- and post-conditions. We chose UML and OCL as they are the de-facto specification language used in the software industry and it is easy to use and understand. However, limited tool set is available for automated analysis of UML models.

We further demonstrate how Alloy [7,8] can be used for the automated verification of the replication protocol. Alloy is a modeling language capable of expressing complex structural constraints and behavior. It has been used to specify network protocols [9–12] and security polices [13–15]. It has very good tool support in the form of the Alloy Analyzer that translates an Alloy specification into a Boolean formula that is evaluated by embedded SAT-solvers. However, few practitioners have the background needed to develop good Alloy models. We provide an approach by which the UML model can be converted into an Alloy model using UML-to-Alloy transformations described in [16]. This obviates the need for protocol verifiers to understand the Alloy specification language.

The approach described in the paper first derives a UML class model from the data replication protocol, and then uses the UML-to-Alloy transformations to generate the Alloy model. The correct behavior of the model is formalized through a set of constraints. The states satisfying these constraints are referred to as valid, and the other prohibited states are called invalid. The generated Alloy model allows a protocol verifier to check whether a system can move from a valid to an invalid state as result of a sequence of operation calls. If analysis uncovers such a sequence of operation calls then the verifier uses the trace information output by the analysis to help find the source of the errors in the UML class model. The analysis was able to uncover a significant error in the original data replication protocol.

We would like to emphasize at this point that the security and privacy issues related to PDS and/or the broker paradigm is not a contribution of this work. The hardware based security services provided by the SPT include protection against side channel attacks. We assume that the broker services are honest but curious. The security protocols needed by the PDS to interact with the broker have been discussed in our previous works [17,18].

The rest of the paper is organized as follows. In Section 2, we first present the architecture of the SPT and its security properties. We then follow it with a design of the PDS and how security and privacy of PHR data is ensured by a combination of hardware security of SPT and anonymous protocols built into a PDS. In Section 3, we discuss how the information in PDS can be replicated. In Section 4, we provide details on the replication protocol. In Section 5, we illustrate our query processing approach. In Section 6, we prove the feasibility of our approach by demonstrating the prototype and providing experimental results. In Section 7, we provide detailed analysis of the properties of the proposed suite of PDS protocols. In Section 8, we provide the formal specification approach, and in Section 9 we describe the results of the formal analysis. In Section 10 we discuss related work, and in Section 11 we conclude the paper and provide our plans for future work.