A Standard-Driven Communication Protocol for Disconnected Clinics in Rural Areas

Massimiliano Masi, Rosario Pugliese and Francesco Tiezzi
Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze
Viale Morgagni, 65 - 50134 Firenze, Italy
Emails: massimiliano.masi@unifi.it, rosario.pugliese@unifi.it, tiezzi@dsi.unifi.it

***Abstract*—The importance of the Electronic Health Record (EHR), which stores all healthcare-related data belonging to a patient, has been recognized in recent years by governments, institutions, and industry. Initiatives like Integrating the Healthcare Enterprise (IHE) have been developed for the definition of standard methodologies for secure and interoperable EHR exchanges among clinics and hospitals. Using the requisites specified by these initiatives, many large-scale projects have been set up to enable healthcare professionals to handle patients' EHRs. Applications deployed in these settings are often considered safety-critical, thus ensuring such security properties as confidentiality, authentication, and authorization is crucial for their success. In this paper, we propose a communication protocol, based on the IHE specifications, for authenticating healthcare professionals and assuring patients' safety in settings where no network connection is available, such as in rural areas of some developing countries. We define a specific threat model, driven by the experience of use cases covered by international projects, and prove that an intruder cannot cause damages to the safety of patients and their data by performing any of the attacks falling within this threat model. To demonstrate the feasibility and effectiveness of our protocol, we have fully implemented it.**

I. Introduction

In recent years, many eHealth projects have started, with the aim of providing optimum patient care. Many governments are now switching from a paper-based healthcare management to an Electronic Health Record (EHR) based solution. An EHR is a set of sensitive data written in a machine readable format (i.e. Hl7's CDA [1]) containing the healthcare history of a patient, including, e.g., the patient summary, the prescriptions, and the dispensations of specific medicines. Software components deployed in these settings need to transmit EHRs among clinics and hospitals for millions of patients (for example, the EU project epSOS [2] potentially serves 500 million patients). These components must protect the healthcare data against malicious attacks. Indeed, tampering or intrusions during the exchange of an EHR may have a direct impact on the life of the patients. Thus, initiatives like [3] and [1] have been promoted for the development and definition of standard methodologies for the secure and interoperable EHR exchange among clinics and hospitals. Using the requirements specified by these initiatives, many projects have been set up, such as [2], [4], [5], [6], [7]. This is not surprising if we also consider that the EU commission has issued a mandate [8] for enforcing the adoption of EHR systems and the US government has published the Health Insurance Portability and Accountability Act (HIPAA) [9] with a similar aim. However using the state-of-the-art international standards, as required by most of the above projects has a significant drawback: some proposed methodologies have not been sufficiently analyzed from a security point of view (as we show in [10]).

In this paper we tackle the problem of sharing patients' healthcare data among clinics without any connection to the Internet[1]. This is a frequent problem in rural areas of developing countries where no network infrastructure is provided by the institutions. Some examples are clinics located in sub-Saharan regions (e.g. Malawi, Namibia) or South African provinces (e.g. Limpopo, Mpumalanga) that can be reached only by means of hundreds of kilometers of sand tracks, or a caravan that travels along townships with first-aid equipment. By relying on the Service Oriented Computing (SOC) paradigm, the Integrating the Healthcare Enterprise (IHE) [3] initiative[2], promotes standards for the definition of services supporting the exchange of healthcare data and the management of patient identifiers. For such a critical scenario, the IHE board proposes the usage of the Cross Enterprise Document Sharing using Portable Media (XDM) [3]. This specification enables the use of different types of electronic support, like CDs, DVDs, and USB drivers, for the transmission of EHRs. The documents stored in such a way can then be delivered by using a car-transportation system.

Specifically, we focus on authentication of healthcare professionals. User authentication is a basic but crucial task for security in general, and healthcare applications in particular. It is required by almost all healthcare projects to enforce authorization of resources (e.g., to prevent unauthorized access to specific medicines or patient summaries) based on attributes related to the professional's identity (e.g. role and department).

As a first contribution of this paper, we propose a protocol based on XDM that guarantees the correct propagation of an authentication process performed remotely while ensuring safety properties in the treatment of patients' healthcare data. Then we define a threat model where a hypothetical attacker (e.g., [11], [12]) can seriously damage the health of patients

This work has been sponsored by the EU project ASCENS, 257414. The first author has been partially supported by the Company Tiani "Spirit" GmbH.

[1]The solution to this problem in presence of Internet connection is shown in [3] and formally studied in [10].

[2]IHE is an initiative by healthcare professionals and industry that strictly follows such international guidelines as HIPAA and EU commission reports.

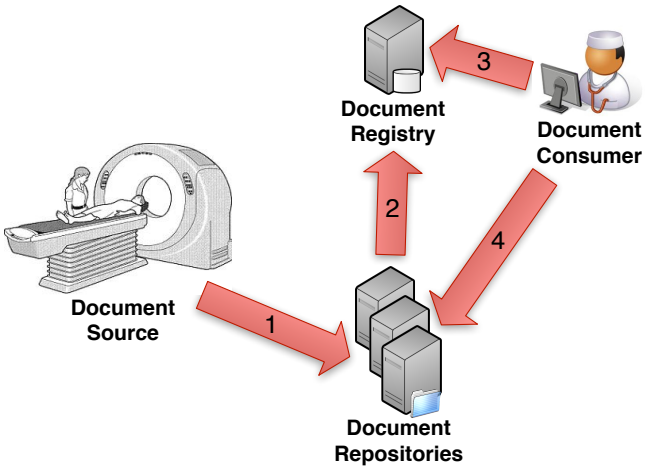

Fig. 1. The XDS workflow

or obtain unauthorized resources by carrying out different types of attacks. As a second contribution, we analyze the protocol to prove that, in our threat model, the attacker cannot perform any action that can compromise the patients' safety. To carry out our investigation, we use a general methodology (see, e.g., [13]) based on formal methods that have been widely used for the verification of complex computer systems (applications to communication protocols are surveyed in, e.g., [14], [15]). In particular, we formally specify the protocol using the process calculus COWS [16] so that the obtained model contains enough details about the involved technologies (this is reflected, e.g., by letting the exchanged messages travel along communication channels with different properties). Then we analyze the obtained model by formulating the protocol's properties as logic formulae and demonstrating their validity through the model checker CMC [17].

It is worth noticing that our protocol has been expressly designed with the aim of guaranteeing *implementability* and at the same time avoiding requiring changes in the scenario under consideration that may have an important impact on the everyday lives of patients or professionals, since most of the projects are already in a production stage.

The rest of the paper is organized as follows. In Section II we provide an account of the scenario with disconnected clinics and introduce our protocol for the authentication of healthcare professionals. In Section III we sketch the formal specification of the protocol and present its analysis. Finally, in Section IV we touch upon comparisons with related work and directions for future work.

II. A SCENARIO WITH DISCONNECTED CLINICS

Modern healthcare applications rely on the use of international standards for exchanging patients' healthcare data. The setting which is used the most for exchanging patient documents within healthcare organizations is based on the IHE Cross Enterprise Document Sharing (XDS) [3] model depicted in Figure 1.

The model uses a central document registry that acts as a catalog for the data. The document source (e.g., a medical device) produces healthcare data for a patient and stores them in one or more document repositories (e.g., databases). The repositories extract the metadata from the documents and update the registry. The document consumer (e.g., a doctor's workstation) queries the registry and obtains the links to the repositories where the data can be downloaded. In a real scenario, dozens of consumers and dozens of sources belong to the same registry.

Each communication in the model is performed by exploiting the SOC paradigm. Each actor (i.e., registry, repository, consumer, source) is a service exchanging SOAP messages with other actors. To provide access control to patients' data, the human requester must be exactly identified. The IHE model defines the use of an authentication assertion, encoded using the Security Assertion Markup Language (SAML) [18], containing the identity of the user sitting behind the client actors. A SAML assertion is a signed security token encoded using the XML format issued by a trusted third party service, the Security Token Service (STS)[3], that contains statements about an authentication procedure performed by an underlying authentication mechanism (such as Kerberos [19]) for a subject. The SAML token is then used by the service requester (in our case, the document consumer) to interact with the services listed in the assertion. The contacted service provider (in our case, the document registry) uses the assertion to authenticate the requester by verifying the digital signature of the trusted issuer.

We consider here a healthcare scenario where the clinics involved need to communicate with hospitals and points of care but network connectivity is missing. This is a common scenario for developing countries where the institutions do not provide access to the Internet in all the regions. IHE addresses this scenario by defining the XDM integration profile. Basically this profile is an XDS model where some communications among document consumers, sources, registries, and repositories are made by writing data to *portable media* (e.g., CDs, DVDs, USB mass storage) and sending them through a car-transportation system.

The outlined scenario presents severe security problems. For example, a patient could forge his own portable media and include new prescriptions for drugs or an intruder could easily gain access to the data by hijacking the carrier on its way (or the carrier itself could act as an intruder).

In Section II-A we present our XDM-based communication protocol which is specifically devised for the setting sketched above. Then, in Section II-B we define a threat model[4] and in Section II-C we explain the need of an extra actor in the protocol to deal with a specific kind of attack considered by our threat model. In the next section we will prove that an intruder cannot endanger patients by performing any of the attacks falling within our threat model.

[3]For the sake of simplicity, we assume an STS that is directly able to authenticate users; i.e., it also plays the role of the identity provider.

[4]It is worth noticing that the major healthcare initiatives are in the way of identifying some specific threat models (see e.g. [20]). We define our threat model using their experience as a basis.

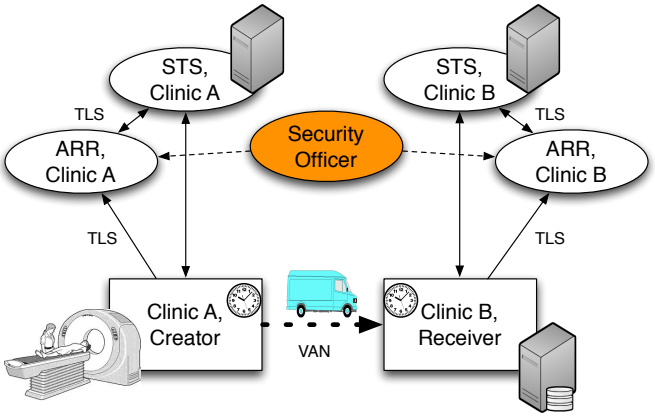

Fig. 2. A typical healthcare scenario with disconnected clinics

$$
\begin{array}{ll}
0. & \left\{ \begin{array}{ll} A \leftrightarrows STS_A & : \ldots \textit{ obtains a SAML assertion} \\ & \quad \textit{for } B \textit{ at } ts1 \textit{ with context } ctx \ \ldots \\ STS_A \rightarrow_{TLS} ARR_A & : ts1, ctx, A, B \\ A \leftrightarrows STS_A & : \ldots \textit{ obtains the} \\ & \quad token(K_A^+, user, \{ctx\}_{K_B^+}) \ \ldots \end{array} \right. \\
1. & A \rightarrow_{TLS} ARR_A : ts1, ctx, A, B, ts2, \{[K_A^+, user, \text{`}doc\text{'}]\}_{K_A^-} \\
2. & A \rightarrow_{VAN} B : A, B, msgId_1, ts2, \{[K_A^+, user, \text{`}doc\text{'}]\}_{K_A^-}, \\
 & \qquad token(K_A^+, user, \{ctx\}_{K_B^+}) \\
3. & B \rightarrow_{TLS} ARR_B : ts2, ts3, A, ctx, \{[K_A^+, user, \text{`}doc\text{'}]\}_{K_A^-}
\end{array}
$$

TABLE I
AN ABSTRACT DESCRIPTION OF THE PROPOSED XDM-BASED PROTOCOL

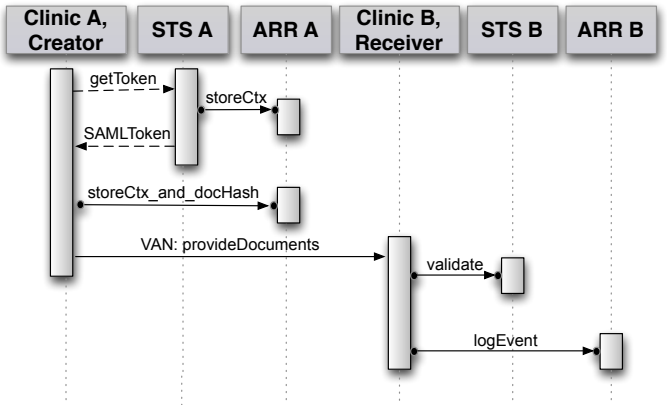

Fig. 3. The scenario with disconnected clinics

A. The abstract model

In the XDS model depicted in Figure 1 (steps 1, 3 and 4) the actors are typically required to establish a secure connection by means of TLS using the Internet. Since we suppose that no Internet connection is available between the consumer/source and the registry/repository, we integrate XDS with XDM, a profile specifically devised for dealing with such settings. In this way, the requirement for having strong authentication in XDS transactions (i.e., a matching X509 certificate between the TLS handshake and the key inside the token, as in [10]) can be relaxed. Indeed, the channel representing the car-transportation system is able to transport only one message in only one direction, hence the TLS handshake cannot be performed.

We abstract the scenario from a logical point of view as depicted in Figure 2. We identify two systems running an XDS suite: clinic A acting as the *creator* and clinic B acting as the *receiver*. The creator runs an instance of a document consumer or a document source, while the receiver runs an instance of a document registry or a document repository. The creator queries or submits documents using the car-transportation system. Each clinic owns an STS that is responsible for issuing and validating SAML tokens. It is worth noting that we make explicit another actor defined by IHE, the ATNA *Audit Record Repository* (ARR), a tamperproof storage that contains the audit trails (i.e., log entries) for each transaction performed by the creator/receiver. We suppose that clinics have local clocks, but do not synchronise.

The model does not abstract away the kind of communication channels used, in order to take into account the different guarantees they provide. Hence, we use here three different kinds of channels. The first one, denoted by $\rightarrow$, is a plain TCP/IP channel. The second one, $\rightarrow_{\mathrm{TLS}}$, is a TCP/IP channel where TLSv1 is available. This means that the confidentiality and integrity of TCP packets are guaranteed. The third one, $\rightarrow_{\mathrm{VAN}}$, represents the car-transportation system conveying patients' healthcare documents. Since this channel only permits sending one message per protocol run, it does not satisfy any mutual authentication properties [21].

The protocol that we propose, written in a notation commonly used to describe security protocols, is shown in Table I and is graphically depicted in Figure 3. The arrow $\longrightarrow$ represents a normal TCP/IP channel, while $\bullet\!\!\longrightarrow\!\!\bullet$ represents the TCP/IP channel where the communication is protected by means of a mutual authenticated TLSv1 implementation. The notation $\{\mathrm{M}\}_{K_B^+}$ stands for the encryption of M using the B's public key and $\{[\mathrm{M}]\}_{K_A^-}$ stands for the signature of M using A's private key (where $[\mathrm{M}]$ is the hash code of M). Each message M represents a SOAP 1.2 message strictly defined by international standards [1], [3], [22]. $ts1$, $ts2$ and $ts3$ are timestamps.

According to the XDM specification, the creator starts by obtaining a token through its local STS. In our setting the token, which for the sake of readability is denoted by $token(K_A^+, user, \{ctx\}_{K_B^+})$, is a SAML assertion. This notation stands for the signed tuple

$$\{[K_A^+, STSname, samlTs, user, B, \{ctx\}_{K_B^+}]\}_{K_{STS_A}^-}$$

containing the name $user$ of the user that is performing the action and where it comes from (indicated by A's public key), the name $STSName$ of the issuer, the timestamp $samlTs$ of the assertion, the receiver B where the assertion can be used, and the encrypted context. Notably, a token permits only one operation via the portable media. The token issuance process is assumed to be performed according to [10] and is not detailed

here for the sake of simplicity (for this reason, it is indicated as step 0 in the protocol). It is, however, worth noticing that a one-to-one correspondence exists between the token issued during the process and the WS-Trust [23] context ctx, that is a unique identifier computed by the STS. The issuance process is also recorded in ARR_A by storing the quadruple $ts1, ctx, A, B$. This quadruple records that a token was issued with context ctx at timestamp $ts1$ for a message sent from A to B. We assume that this message is sent over $\rightarrow_{TLS}$.

In message 1, clinic A updates the log in ARR_A by adding a new timestamp $ts2$, representing the instant when the portable medium is created (e.g. the CD is burned), and a signed tuple, representing the username of the physical person sitting in front of the workstation, which is the same as the subject of the SAML assertion, and the document, here abstracted as 'doc'. Notably, we are not deviating from the standards. In fact, the username value is part of the metadata accompanying the document itself; the interested reader is referred to the XDS.b documentation (*XCN values*, [3]) for the other metadata.

In message 2, clinic A sends the portable medium through the car-transportation system that travels to B. The message contains a unique identifier, $msgId_1$, the signed document and the SAML token.

When clinic B receives the message, it checks if the value $user$ equals the subject of the assertion and if the key contained in the signed document matches the holder-of-key data inside the token. If both tests succeed, clinic B validates the assertion to its STS_B through message 3,, as in [10], and writes in its local ARR_B the following knowledge: "someone, probably A, at time $ts2$, sent a message with identifier ctx for a document represented by the signed tuple $\{[K_A^+, user, \text{'}doc\text{'}]\}_{K_A^-}$; now is $ts3$". The document also contains the certificate that corresponds to the signer's public key.

B. The threat model

By taking the features of the different kinds of communication channels into account, we consider a standard intruder *à la* Dolev-Yao acting only along channel $\rightarrow_{VAN}$. The intruder can see all the messages passing through the channel (i.e., he can read the documents delivered via van), he or she is a legitimate user of the network (in our setting this means that he or she is able to sign documents) and has the opportunity to be a receiver for any user. We also assume that encryption and hashing functions cannot be broken.

Our assumption that channels of type $\rightarrow_{TLS}$ are not subject to the intruder is supported by the fact that, in the real world, each clinics' information system usually runs on a single and audited, machine. Therefore the problem of intercepting messages can be solved better by using intrusion detection techniques.

We thus identify four different types of attacks the intruder I can perform:

1) I suppresses a message. For instance, I could suppress a submission for a new rare form of allergy or a query for crucial healthcare data and, in both cases, the suppression may lead to the death of the patient if the patient travels among clinics. It is thus crucial for patients' safety to consider suppressed messages as attacks, as these messages could bear data necessary for the patients' lives. Moreover, since messages traveling along $\rightarrow_{VAN}$ may have unbounded delays, to stay on the safe side, we also consider these delayed messages to be attacks. To appropriately deal with these attacks we introduce a new actor, called *Security Officer* (see Figure 2) in the scenario. This actor's role will be detailed in the next section.
2) I acts as a healthcare professional sitting at A wanting to access restricted resources by reusing a previously issued SAML token. For instance, suppose that $nurse$ (a valid user at A) wants to obtain a large amount of drugs for the illegal market. He or she could reuse an assertion, which was issued for another purpose and has already been used to create a new illicit prescription. $nurse$ then steals the assertion from the portable medium, attaches it to his or her new prescription and sends:
$$A \rightarrow_{VAN} B: \quad \begin{array}{l} A, B, msgId, ts2, \\ \{[K_A^+, nurse, \text{'}prescr\text{'}]\}_{K_A^-}, \\ token(K_A^+, user, \{ctx\}_{K_B^+}) \end{array} \qquad (1)$$
This attack is discovered by the $Receiver$ because the user of the token is different from the creating user embedded in the document.
3) I obtains the message by listening on $\rightarrow_{VAN}$, suppresses it, and sends:
$$I \rightarrow_{VAN} B: \quad \begin{array}{l} I, B, msgId, ts2, \\ \{[K_A^+, user, \text{'}prescr\text{'}]\}_{K_I^-}, \\ token(K_A^+, user, \{ctx\}_{K_B^+}) \end{array} \qquad (2)$$
The difference between this attack and attack number 2 is that there *nurse* is sitting at clinic A, while here the intruder is intercepting the van. This attack is discovered by the $Receiver$ because the public part of the intruder's signing key K_I^- is different from the key associated with the document and inside the token.
4) I sends the same (intercepted) message multiple times, for example to obtain the same resource multiple times. This attack is similar to attacks 2 and 3, and is a form of replay attack. It is discovered by the $Receiver$ because multiple ctx are present in the database (only the message which is received first is left).

C. The role of the officer

Let us now suppose there are several clinics $A_1, \ldots, A_n$, such that each A_i has its own ARR_i and acts as a creator for the clinic B. The officer polls the clinics in a round-robin fashion to detect if attack 1 has been performed, in which case he or she establishes which actions must be performed by the clinics as countermeasures. To this aim, every quantum of time t the officer checks the logs of every ARR_i by appropriately comparing them with the logs stored in the clinic B.

When the officer visits clinic A_i, he or she obtains the set $\mathcal{A}_i^{log}$ of all the audit trails from his or her last visit. Let m_i be the maximum among the timestamps of the audits (the officer saves the previous value of m_i as $oldm_i$). When the officer visits clinic B, he or she obtains the set $\mathcal{B}^{log}$ of all the audits trails. Let's recall what an audit located in ARR_B looks like. It can be defined as a tuple $\mathbf{a} = \langle creation\_ts, arrival\_ts, sender, ctx, Signature \rangle$, where $creation\_ts$ is the creation time at $sender$, $arrival\_ts$ is the arrival time at B, ctx is the context, and $Signature$ is the signature of the document. We use $\mathbf{a}(1)$ to denote $creation\_ts$ and $\mathbf{a}(3)$ to denote $sender$. The officer can then partition $\mathcal{B}^{log}$ by defining:

$$\mathcal{B}^{log}|_{A_i,m_i} = \left\{ \mathbf{a} \mid \mathbf{a}(3) = A_i \wedge oldm_i < \mathbf{a}(1) \leq m_i \right\}$$

Each element $\mathcal{B}^{log}|_{A_i,m_i}$ of the partition contains the audits coming from clinic A_i with a timestamp greater than the old reading $oldm_i$ taken by the officer and not greater than the last timestamp m_i read at A_i.

The officer can now tell the different situations apart. If $\mathcal{A}_i^{log} = \mathcal{B}^{log}|_{A_i,m_i}$ then all messages produced at clinic A_i arrived safely at clinic B. If this holds for all clinics, then we have $\bigcup_{i=1}^{n} \mathcal{A}_i^{log} = \mathcal{B}^{log}$ which means that every message from any clinics has safely arrived at B. If instead $\mathcal{A}_i^{log} \backslash (\mathcal{B}^{log}|_{A_i,m_i}) \neq \emptyset$ then there are messages in A_i that have not yet arrived at B. The messages are suppressed or are late (attack 1). In the opposite situation, i.e., when $\mathcal{B}^{log}|_{A_i,m_i} \backslash \mathcal{A}_i^{log} \neq \emptyset$, then there are more messages at B than those produced by the clinics, thus some messages have been introduced into the channels by the intruder (attacks 2, 3 and 4).

III. Specification and analysis

Our formal analysis is based on the use of the process calculus COWS [16] to specify the protocol and the transactions involved, while reflecting many real-world implementation details (e.g. algorithms, field names and message exchanges are taken from OASIS standards). Due to their rich repertoire of elegant meta-theories, proof techniques and analytical tools, process calculi currently play a central role in laying rigorous methodological foundations for specification and validation of SOC applications. Our preference for COWS is motivated by its mechanisms and primitives that have proven to be particularly expressive for modeling the behavior of service-oriented applications (see, e.g., [24], [25]). Moreover, the calculus is equipped with tools for formulating properties of COWS specifications and demonstrating their validity, like the temporal logic SocL and its model checker CMC [17] that we use in our investigation. In fact, process calculi together with modal and temporal logic, possibly supported by efficient software tools, have long been proved suitable to reason about the design of complex computing systems (see, e.g., [26], [27], [28]).

For the sake of simplicity, we present here just a sketch of the COWS specification of the protocol. We refer the interested reader to [29] for a complete account of the specification, and to [16], [25] for the presentation of COWS and of many examples illustrating its peculiarities and expressiveness.

The COWS term representing the overall scenario is

```
let ... in
  [hashReq♯] [hashResp♯] ...
  ( Sha(hashReq,hashResp)
    | Cipher(...)
    | Signer(...)
    | ClinicA(hashReq,hashResp,...)
    | ClinicB(hashReq,hashResp,...)
    | Officer(...) )
  | ... intruders ...
end
```

where to make the reading easier, we have omitted irrelevant details. The protocol has three main participants, `ClinicA`, `ClinicB` and `Officer`. They are composed by using the parallel composition operator _|_ that allows them to be concurrently executed and to interact with each other. Since COWS does not offer primitives for, e.g., *encryption* and *hashing*, these and other useful security-related features are provided to each participant through a library of functions, implemented as a set of shared services. We have implementations of such algorithms as SHA for hashing (`Sha`), RSA for public-key cryptography (`Cipher`), and digital signatures (`Signer`), that are necessary to properly manage the data to be exchanged during protocol runs. These COWS services play a role similar to that of functions in the applied π-calculus [13]. The delimitation operator [_]_ is used here to restrict access to services `Sha`, `Cipher`, and `Signer` by declaring that `hashReq`, `hashResp`, ... are private operation names known only to the three participants (other than, of course, the services that provide them). As usual, the `let` construct permits the definition of specifications in a modular style.

Within the above `let` construct, the COWS service definition of the creator clinic is:

```
ClinicA(hashReq,hashResp,...) =
  [clockA#][write#][getToken#] ...
  ( Clock(clockA,...) | ARR(write,...) | STS(getToken,,...)
    | Creator(clockA,write,getToken,...) )
```

The term `ClinicB` representing the receiving clinic is similar to `ClinicA`, but for the term `Receiver` in place of `Creator`. Each clinic has its own local `Clock`, `ARR` and `STS`, with which it shares private partner and operation names (e.g., `clockA`, `write` and `getToken`). These names permit defining private endpoints to simulate internal interaction with `Clock`, secure connections with `ARR` along channels of the type $\rightarrow_{TLS}$, and authenticated connections with the `STS` along channels of the type $\rightarrow$ (as dealt with in [10]). The local clock ticks along a private endpoint and, when prompted, returns the current value to the requester. If the `STS` receives a request from the associated participant instead, then it generates and returns a SAML token containing a unique context. Finally, `ARR` instantiates a *stack*, i.e., a LIFO data structure used to store audit trails, and provides different functionalities depending on the role of the participant. In the case of a creator role, `ARR` waits the first message from the `STS` stating that an audit trail with a given context will arrive from the clinic. When

the audit is received, it is pushed onto the stack. In the case of a receiver role, ARR simply pushes the received audit trail onto the stack, unless the audit trail contains a context already stored (attack 4).

The security Officer is in charge of revealing attack 1, which occurs when the intruder suppresses messages. Only one instance of Officer can run at a given time (in order to reduce the model state space). Once activated, Officer first gets the values stored by Creator's ARR and saves the maximum timestamp of the audits in the Creator's stack in a variable MaxT1. Then, Officer gets the audits, which have timestamp not greater than MaxT1, stored by Receiver's ARR. After collecting the audits from the two repositories, Officer compares them to detect an attack 1, as defined in Section II-C.

Each of the four attacks described in Section II-B is modeled as a COWS term running in parallel with the protocol's participants. As expected, the intruder can freely suppress/forge/read messages along the $\rightarrow_{VAN}$ channel. Each attack is revealed by evaluating a SocL formula that exploits actions signaling that specific data tuples are sent along some given endpoints.

For instance, the COWS term modeling the intruder conducting attack 1 is

```
[A][MsgId1][Ts2][SignedDoc] ...
 b.van?<A,b,MsgId1,Ts2,SignedDoc,...>.
 (sys.attack1!<messageSuppressed,SignedDoc>
  | sys.attack1?<messageSuppressed,SignedDoc>.
      officer.activate!<> )
```

The intruder starts with a receive activity of the form b.van? $<$ A, b, MsgId1, Ts2,SignedDoc, ... $>$ corresponding to the reception of a message for B over the channel $\rightarrow_{VAN}$ (here rendered as the public endpoint b.van). The receive activity initializes the variables A, MsgId1, Ts2, SignedDoc,..., declared local to the term by the delimitation operator, with the message data. When the intruder intercepts a message, communication along the endpoint sys.attack1 takes place (i.e., the invoke activity sys.attack1! $<$ messageSuppressed, SignedDoc $>$ and the receive activity sys.attack1? $<$ messageSuppressed, SignedDoc $>$ synchronise); this is used during the analysis to signal that the system is under attack. Only at this point Officer is activated (by means of a signal over officer.activate), after which it can go to clinic A and B at any time.

The analysis of the protocol is carried out by exploiting the software tool CMC for model checking SocL formulae over COWS specifications. SocL [17] is an action- and state-based, branching time, temporal logic specifically designed to express properties of service-oriented systems. We will not present all the constructs of SocL (we refer the interested reader to [17] for a detailed presentation of the logic and its applications), but only explain the formula used to reveal an attack 1:

```
AG [attack1Performed(message,$doc)]
   AF {attack1DetectedByOfficer(message,%doc)} true
```

The formula means that it holds *globally* (AG), i.e., in any state of the model, that *if* (operator $[\ldots]$[5]) an attack of type 1 is performed by the intruder, then *always* (AF) this attack will be detected by Officer.

The previous formula is stated in terms of *abstract* actions, meaning that, e.g., an attack of type 1 has been performed, while the COWS specification is stated in terms of *concrete* actions, i.e. communication of data tuples along endpoints. To verify an abstract property over a concrete specification, CMC permits specification of a set of *abstraction rules*. For example, rule

```
Action sys.attack1<messageSuppressed,$signedDoc>
       -> attack1Performed(message,$signedDoc)
```

maps the concrete action corresponding to a communication along sys.attack1 to the abstract action attack1Performed. A similar rule is used for mapping a communication along sys.attack1Detected, performed by Officer when it discovers that a message has been suppressed, to the abstract action attack1DetectedByOfficer. Evaluation of the above formula returns TRUE, which means that if a message containing a certain document is suppressed, then the officer will discover that B did not receive that specific message.

The use of abstraction rules to relate concrete actions of specifications with abstract actions of formulae is a peculiarity of CMC and of the verification methodology it enables. This permits one to initially formalize the desired properties as SocL formulae in a way that is independent from individual specifications and then to verify them by tailoring the specification under analysis by means of proper abstraction rules.

The remaining attacks are detected by following a similar approach; for each attack we demonstrate that the intruder cannot successfully perform it. We refer the interested reader to [29] for the complete description of the analysis.

IV. Conclusions

We have presented a feasible and effective communication protocol, based on international standards, for exchanging patient healthcare information among disconnected clinics and hospitals, while preserving the security of healthcare data and the safety of patients.

We have considered the IHE standards based on the XDS family [3] for authenticating healthcare professionals in transactions related to exchange of patients' data. In particular, to deal with a scenario where clinics do not have an Internet connection and exchange data by using a car-transportation system, we rely on the IHE XDM integration profile. We defined a specific threat model driven by the experience of use cases covered by international projects. Then we formalized our protocol using the process calculus COWS and analyzed it through the model checker CMC to prove that it is robust enough to hold out against the attacks described in Section II. It is worth noting that our protocol is password-based in order to comply with the standard WS-Security Username

[5]This is the modal logic operator *box*: [a]f states that, no matter how a process performs action a, the state it reaches in doing so will *necessarily* satisfy the property expressed by f.

Token Profile. Therefore, the protocol is secure as long as the passwords are kept secured.

Since our reference standards are well-understood and implemented, we set ourselves the goal of only suggesting changes that can be easily implemented by vendors. Indeed, we have implemented the protocol using WS-Trust 1.3, SAML 2.0, WS-Security and WS-Security Username Token Profile 1.1. We have also used the Axis2 libraries[6] and the JBoss application server[7]. Our Java implementation consists of four services: a *Document Consumer*, a *Document Registry*, a *Document Repository* and a *Security Token Service*[8]. Moreover, according to the XDM standard, an application capable of writing messages in the portable media is exploited. All the XDS and XDM services are given as a courtesy of the company Tiani "Spirit" GmbH[9], located in Vienna, Austria. An outline of the overall implementation of the protocol is available online [29].

A. Related work

Analysis of security protocols using formal methods is not a novel research field (see, e.g., [14] and [15] for a survey). For what concerns analysis of web services, Microsoft Research proposes the specification languages TulaFale [30], [31] and TLA+ [32]. TulaFale uses the model checking engine CryptoVerif [33] and only considers SOAP message rewrite attacks. Instead, our threat model covers other kinds of attacks whose relevance in healthcare scenarios emerges from the experience gained from international projects. We leave the study of the interplay between rewrite attacks and the use of the SAML specification for future investigation, since, to the best of our knowledge, any SAML-based protocol could be subject to rewrite attacks. In [30] the authors analyze WS-Trust for a secure exchange of a Security Context Token while we consider WS-Trust for issuing a SAML token. TLA+ is a language based on the Temporal Logic of Actions whose specifications can be analysed by using the model checker TLC [34]. In [35] the authors analyze the Web Services Atomic Transaction protocol, which however is a protocol for distributed transactions among web services rather than a security protocol.

The Casper tool [36] gives the possibility to define properties of the communication channel. However, Casper's main focus is in proving the hierarchy of Lowe's authentication properties [21] which do not hold in the case of our channel representing the car-transportation system.

The SAML 1.0 and 2.0 specifications have been studied in e.g. [37], [38], [39]. However, these works concentrate on the SAML Protocol and Profiles [40] to obtain SAML Authentication assertions, while we focus on WS-Trust. The work closest to ours is [37] where the SAML-based Single Sign-On for Google Apps is analyzed with the tool AVISPA [41]. A flaw in the Google implementation is found, where a fake service provider can potentially access a Google resource without knowing the password of the user. The flaw discovered is in the format of the SAML assertion, which lacks the `Audience` list. In our scenario, however, this kind of attack cannot occur since the `Audience` list must be contained in the assertion and refer to the registry.

B. Future work

As the previously mentioned related work and ours witness, designing communication protocols that simply adhere to IHE specifications does not guarantee the absence of security flaws. Due to the widespread diffusion of such standards, especially for managing EHRs, it is thus worthwhile to pursue our formal methods-based investigation of the security issues that can arise in healthcare environments. An issue that needs to be considered is mobility of healthcare professionals among clinics and hospitals. In fact, differently from patients, healthcare professionals need to authenticate themselves, possibly on different clinics. Since IHE does not provide any standard for regulating such mobility, to guarantee the coherence of ARR's audit trails some commonly used solutions can be exploited, ranging from using different accounts for the same professional to adopting a synchronized directory structure. However, due to the lack of Internet connectivity in our setting, feasibility of these solutions needs further investigation. Moreover, we plan in the near future to study the application of the least-privilege concept to the automated enforcement of XACML-based access control policies.

[6]Available at http://ws.apache.org/axis2.

[7]Available at http://www.jboss.org.

[8]Available as Axis2 service at http://178.188.229.34:41081/SpiritIdentityProvider/services/STS09?wsdl.

[9]Web site: http://www.tiani-spirit.com

REFERENCES

[1] Health Level Seven organization, "Hl7 standards," 2009, http://www.hl7.org.

[2] The epSOS project, "An European eHealth Project," 2010, http://www.epsos.eu.

[3] The IHE Initiative, "IT Infrastructure Technical Framework," 2009, http://www.ihe.net.

[4] The Nationwide Health Information Network (NHIN), "An American eHealth Project," 2009, http://healthit.hhs.gov/portal/server.pt.

[5] GIP DMP, "Dossier Médical Personnel," 2009, http://www.d-m-p.org.

[6] ARGE-ELGA, "Die österreich elektronische gesundheitsakte," 2008, http://www.arge-elga.at.

[7] The South African Department of Health, "EHR project in South Africa," 2009, http://southafrica.usembassy.gov/root/pdfs/pepfar-hmis-docs/ndoh-e-hr-for-south-africa.pdf.

[8] EU Commission, "M/403 EN: Standardisation mandate addressed to CEN, CENELEC and ETSI in the field of Information and Communication Technologies," European Commission Enterprise And Industry Directorate-General, Tech. Rep., 2007, http://ec.europa.eu/enterprise/standards_policy/mandates/database/index.cfm?fuseaction=search.detail&id=363#.

[9] US Congress, "Health Insurance Portability and Accountability Act," Department of Health, Tech. Rep., 1996, http://www.cms.gov/HIPAAGenInfo/.

[10] M. Masi, R. Pugliese, and F. Tiezzi, "On Secure Implementation of an IHE XUA-Based Protocol for Authenticating Healthcare Professionals," in *ICISS*, ser. LNCS, vol. 5905. Springer, 2009, pp. 55–70.

[11] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–207, 1983.

[12] P. Broadfoot and G. Lowe, "On Distributed Security Transactions that Use Secure Transport Protocols," in *CSFW*. IEEE Computer Society, 2003, p. 141.

[13] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," in *POPL*. ACM, 2001, pp. 104–115.

[14] L. Ma and J. Tsai, "Formal verification techniques for computer communication security protocols," *Handbook of Software Engineering and Knowledge Engineering*, vol. 1, pp. 23–46, 2001.
[15] C. Fidge, "A Survey of Verification Techniques for Security Protocols," Software Verification Research Centre, The University of Queensland, Tech. Rep. 01-22, 2001.
[16] A. Lapadula, R. Pugliese, and F. Tiezzi, "A Calculus for Orchestration of Web Services." in *ESOP*, ser. LNCS, vol. 4421. Springer, 2007, pp. 33–47.
[17] A. Fantechi, S. Gnesi, A. Lapadula, F. Mazzanti, R. Pugliese, and F. Tiezzi, "A model checking approach for verifying COWS specifications," in *FASE*, ser. LNCS, vol. 4961. Springer, 2008, pp. 230–245.
[18] OASIS Security Services TC, "Assertions and protocols for the OASIS security assertion markup language (SAML) v2.02," 2005, http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf.
[19] B. Neuman and T. Ts'o, "Kerberos: An authentication service for computer networks," *IEEE Communications Magazine*, vol. 32, no. 9, pp. 33–38, 1994.
[20] The Nationwide Health Information Network (NHIN), "Threat models," 2009, http://wiki.directproject.org/Threat+Models.
[21] G. Lowe, "A Hierarchy of Authentication Specifications," in *CSFW*. IEEE Computer Society, 1997, pp. 31–44.
[22] W3C, "World Wide Web Consortium," http://www.w3.org.
[23] OASIS Web Services Security TC, "WS-Trust 1.3," 2007, http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.pdf.
[24] A. Lapadula, R. Pugliese, and F. Tiezzi, "Specifying and Analysing SOC Applications with COWS," in *Concurrency, Graphs and Models*, ser. LNCS, vol. 5065. Springer, 2008, pp. 701–720.
[25] F. Tiezzi, "Specification and Analysis of Service-Oriented Applications," PhD Thesis in Computer Science, Dipartimento di Sistemi e Informatica, Università degli Studi di Firenze, 2009, http://rap.dsi.unifi.it/cows/theses/tiezzi_phdthesis.pdf.
[26] E. M. Clarke, O. Grumberg, and D. Peled, *Model Checking*. MIT Press, 1999.
[27] J. Bradfield and C. Stirling, "Modal logics and mu-calculi: an introduction," *Handbook of Process Algebra*, pp. 293–330, 2001.
[28] O. Grumberg and H. Veith, Eds., *25 Years of Model Checking - History, Achievements, Perspectives*, ser. LNCS, vol. 5000. Springer, 2008.
[29] M. Masi, R. Pugliese, and F. Tiezzi, "Security analysis of standards-driven communication protocols for healthcare scenarios," Università degli Studi di Firenze, Dipartimento di Sistemi e Informatica, Tech. Rep., 2010, http://rap.dsi.unifi.it/cows/papers/xds-xdm.pdf.
[30] K. Bhargavan, R. Corin, C. Fournet, and A. Gordon, "Secure sessions for web services," in *SWS*. ACM, 2004, pp. 56–66.
[31] K. Bhargavan, C. Fournet, A. Gordon, and R. Pucella, "TulaFale: A Security Tool for Web Services," in *FMCO*, ser. LNCS, vol. 3188. Springer, 2004, pp. 197–222.
[32] L. Lamport, *Specifying Systems, The TLA+ Language and Tools for Hardware and Software Engineers*. Addison-Wesley, 2002.
[33] B. Blanchet, "CryptoVerif: Computationally Sound Mechanized Prover for Cryptographic Protocols," Dagstuhl seminar "Formal Protocol Verification Applied", 2007.
[34] L. Lamport and Y. Yu, "TLC – The TLA+ Model Checker," 2003, http://research.microsoft.com/en-us/um/people/lamport/tla/tlc.html.
[35] J. Johnson, D. Langworthy, L. Lamport, and F. Vogt, "Formal specification of a web services protocol," *J. Log. Algebr. Program.*, vol. 70, no. 1, pp. 34–52, 2007.
[36] G. Lowe, "Casper: A Compiler for the Analysis of Security Protocols," *Journal of Computer Security*, vol. 6, no. 1-2, pp. 53–84, 1998.
[37] A. Armando et al, "Formal Analysis of SAML 2.0 Web Browser Single Sign-On: Breaking the SAML-based Single Sign-On for Google Apps," in *FMSE*. ACM, 2008, pp. 1–10.
[38] T. Groß, "Security analysis of the saml single sign-on browser/artifact profile," in *ACSAC*. IEEE Computer Society, 2003, pp. 298–307.
[39] S. Hansen, J. Skriver, and H. Nielson, "Using static analysis to validate the SAML single sign-on protocol," in *WITS*. ACM, 2005, pp. 27–40.
[40] OASIS Security Services TC, "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0," 2005, http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf.
[41] A. Armando et al, "The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications," in *CAV*, ser. LNCS, vol. 3576. Springer, 2005, pp. 281–285.

List of acronyms

ARR: Audit Record Repository
ATNA: Audit Trail and Node Authentication
AVISPA: Automated Validation of Internet Security Protocols and Applications
CDA: Clinical Document Architecture
CMC: COWS model checker
COWS: Calculus for Orchestration of Web Services
EHR: Electronic Health Record
epSOS: European Patients Smart Open Services
HIPAA: Health Insurance Portability and Accountability Act
Hl7: Health Level Seven International
IHE: Integrating the Healthcare Enterprise
IP: Internet Protocol
LIFO: Last In First Out
OASIS: Organization for the Advancement of Structured Information Standards
SAML: Security Assertion Markup Language
SHA: Secure Hash Algorithm
SOAP: Simple Object Access Protocol
SOC: Service-Oriented Computing
SocL: Service-Oriented Computing Logic
STS: Security Token Service
TCP: Transmission Control Protocol
TLA+: Temporal Logic of Actions specification language
TLC: TLA+ model Checker
TLS: Transport Layer Security
W3C: World Wide Web Consortium
XACML: eXtensible Access Control Markup Language
XCN: eXtended Composite ID Number and name for persons
XDM: Cross Enterprise Document Sharing using Portable Media
XDS: Cross Enterprise Document Sharing
XML: eXtensible Markup Language