On the usage of SAML delegate assertions in an healthcare scenario with federated communities

Massimiliano Masi Roland Maurer

May 25, 2010

Abstract

The importance of the Electronic Health Record (EHR) has been addressed in recent years by governments and institutions. Many large scale projects have been funded with the aim to allow healthcare professionals to consult patients data amongst different organizations. Concepts like interoperability, security and confidentiality are the key for the success for these projects. The Integrating the Healthcare Enterprise (IHE) initiative promotes the coordinated use of established standards for authenticated and secure EHR exchange amongst clinics and hospitals or even regions. For these scenarios, the problem of having authenticated transactions is crucial, in order to provide a form of authorization while accessing patient healthcare information. The IHE initiative addresses the problem by means of SAML assertions, i.e. XML documents containing authentication statements. In this paper, we focus on the problem of propagating the authentication information of healthcare professionals amongst hospitals or regions (in the IHE jargon, *communities*) by relying on the delegation mechanism introduced by SAML.

1 Introduction

The secure exchange of patient healthcare information amongst clinics and hospitals is becoming a crucial task due to the numerous initiatives introduced by governments and institutions in recent years [1, 2, 3, 4, 5].

The Integrating the Healthcare Enterprise (IHE) [6] is a worldwide initiative founded for promoting the coordinated use of established standards (see e.g. [7, 8]) to improve information sharing in an healthcare scenario.

In particular, the IHE profiles named XUA, ATNA and BPPC are voted to address the problem of security, authentication and authorization, by exploiting a Service Oriented Computing (SOC) approach and adopting OASIS standards, such as SAML [9], ebXML [10] and WS-Trust [11].

The IHE initiative introduces the concept of *communities*, a set of healthcare facilities such as hospitals and clinics that cooperate together for exchanging healthcare data. Examples of communities are regions, federated hospitals or even countries. A community is uniquely identified by a *gateway*, a software entity responsible for the transmission of data with other communities, as described in the IHE XCA [12] profile.

Healthcare professionals authentication is one of the basic requirements for the access of person related health data. The IHE profile XUA makes use of SAML authentication assertions for propagating the identity of the user. A SAML assertion is a signed XML document issued by the *identity provider*, a service able to attest the identity of an user and to create a digital identity based on authentication made by an underlying mechanism (i.e. Kerberos).

The application of XUA in a cross community scenario introduces issues that a security architect needs to address in order to provide optimum patient care, i.e. for applying access control policies. In order to perform a transaction that crosses the border of a community, the SAML assertion obtained by the local identity provider have to pass through the gateway that "forwards" the authentication claims to the remote services.

However, impersonation and forwarding must be avoided specially in service oriented computing because of its stateless nature. In fact, before performing any cross community transaction, the gateway needs to perform an authentication for supporting the transactions, but *on behalf* of the real user. The informations about the user belonging to the community should be forwarded to the remote community by the gateways. This is covered also by OASIS as *direct brokered trust* [11, 13].

The main contribution of this work is to provide a solution of the problem of using IHE XUA in a cross community scenario (also called *federated* scenario). Our solution is based on the use of SAML *delegate* authentication assertions. We show that our solution does not affect any existing standard and we report our experience with a pilot project running in a Austrian region.

This work is structured as follows: in Section 2 we firstly briefly revise the details of the healthcare security standards, XCA, XUA and SAML. In Section 3 we describe our solution to the problems introduced by the usage of delegate assertions in the healthcare scenario and in Section 4 we touch upon future and related work and we conclude. A full version of this work is

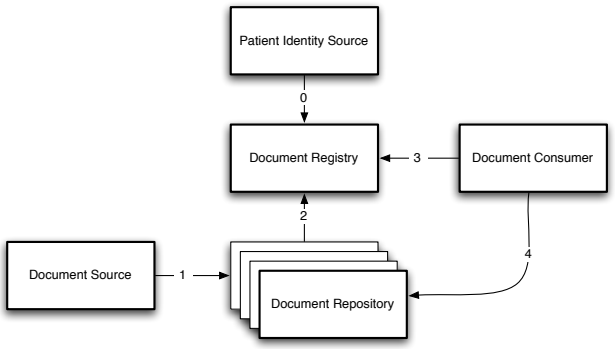

Figure 1: The Cross Enterprise Document Sharing scenario

available online [14].

2 Communities

The basic IHE patient healthcare data exchange pattern is based on an ebXML [10] model, called Cross Enterprise Document Sharing (XDS). This specification consists in four actors, a document *source* that creates documents for the patient (e.g. radiological images), a document *consumer*, a workstation that displays healthcare documents to the professional, a document *repository* that provides storage and finally a document *registry*, the ebXML catalogue for documents metadata.

The workflow is depicted in Figure 1: we suppose that a patient identifier has been created by the Patient Identity Source actor (step 0). In the first step, the source submits documents the repositories. The document contains additional metadata that are used by the repository to push the information about the documents to the registry (step 2). These data may contain the patient identifier, the document type and the sensitivity of data (e.g. confidential, access restrict). The document consumer queries the registry for accessing the data. Registry returns back to the queries with a link for the repository where the documents are effectively stored. The document consumer retrieves then the document and displays it to the healthcare professional (step 3 and 4).

In the IHE model all the services that share the same registry instance are logically grouped as an *affinity domain*. A set of affinity domains that decide to cooperate together for exchanging documents are identified as a *community*. Communities are uniquely identified by a service, called *gateway*, that acts a proxy (i.e. it intercepts all the messages that are travelling amongst communities). It is worth noticing that from a functional point of view a community can be established not only by federating affinity domains but any kind of healthcare related software that uses a gateway for accessing data can be abstracted as a community itself, if the gateway observes the be-

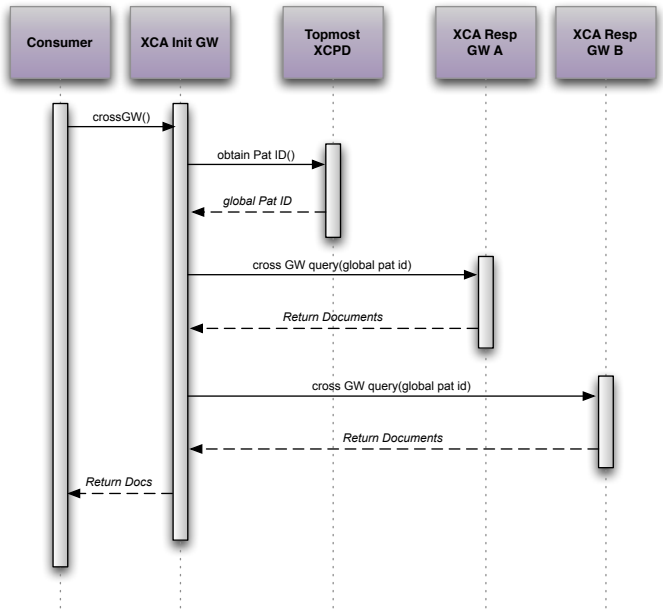

Figure 2: A cross community query

havior defined by [6]. An example of a community built upon a Computing GRID is shown in [15].

When a client service (a document consumer or a source) needs to perform a *cross gateway query* (i.e. to query for a document in a foreign community) it contacts the local gateway (that is called *initiating* gateway) that creates a message to be sent to all the known gateways. The reason for that is due to the fact that it could be unknown in which community the document related to the searched patient is stored.

In order to be able to perform a successful query, the patient identifier is obtained by the initiating gateway using the IHE XCPD mechanism [16]. The Cross Community Patient Discovery (XCPD) profile supports the ability to locate communities which hold a patient's relevant health data and the translation of patient identifiers across communities holding the same patient's data. The XCPD service can be hierarchical: each community can have its own XCPD service consulting the local patient index or it can exist a topmost service able to contact the community's patient indexes. For the sake of simplicity in this work we consider the last option, where a centralized XCPD service contacts the local patient indexes. The complete workflow is depicted in Figure 2.

3 Applying security

The IHE security model is driven by the Audit Trail and Node Specification profile. Each machine containing and transmitting healthcare data must possess an X.509 certificate and a private key attesting the identity. Access to data must be restricted only by using audited XDS transactions secured by means of TLS communication channels. User authentication is achieved by the Cross Enterprise User Assertion profile that defines the usage of a SAML authentication assertion to be used with all XDS transactions.

The Security Assertion Markup Language (SAML) [9] is an OASIS standard that defines the exchange of security statements using signed XML documents called *assertions*, playing the role of security tokens. SAML introduces a new actor, called *identity provider* responsible to attest the identity of an user based on an authentication procedure performed by an underlying authentication mechanism (such as Kerberos) for a *Subject*. The SAML assertion contains an element called `AudienceRestriction` where all the services for which the token is intended are listed. The interaction is made by using of WS-Security [17] for embedding the token in each IHE transaction that is requesting for health data (e.g. steps 1, 3 and 4 in Figure 1). The contacted *service provider* uses the assertion for authenticating the requester. SAML subjects can be confirmed with the method listed in the `SubjectConfirmation` element. It is worth noticing that IHE does not define a structure of the token neither a method for obtaining it by resulting in potential weak and insecure implementation. We assume a secure token issuance process as in [18].

In our federated scenario, the application of the IHE security model is not straightforward. The users firstly authenticates using the local identity provider (located in the healthcare facility, affinity domain or community) for enabling authentication in transactions performed within the community (i.e. an XDS transaction from a document consumer to the registry). When the authentication process begins, the client service needs to specify the list of recipients to be placed in the `AudienceRestriction` element.

It is important to note that at this stage that the identity provider is allowed to encrypt data (e.g. the user identifiers [9], or as in [18] the WS-Trust issuing context identifier). Under these assumptions, the SAML assertion issued by the local identity provider for the usage with the local services cannot be used for performing any cross gateway transaction.

An high level overview of the motivating scenario is shown in the Table 1[1].

[1]In order to easy the readability of the article, at this stage we assume that every message is travelling along secure and authenticated channels. We do not introduce here additional signatures checks that are out of the scope of this work. A more detailed account

$$
\begin{array}{lcll}
A \rightarrow I & : & A, msgId_1, I, \{[RST(user, role, org, B, C)]\}_{K_A^-} & (1) \\
I \rightarrow A & : & I, msgId_2, msgId_1, A, RSTR(\{[I, role, org, \{u_{id}\}_{K_B^+}, \{u_{id}\}_{K_C^+}]\}_{K_I^-}) & (2) \\
A \rightarrow B & : & A, msgId_3, B, \{[I, role, org, \{u_{id}\}_{K_B^+}, \{u_{id}\}_{K_C^+}]\}_{K_I^-}, XDSData & (3) \\
A \rightarrow C & : & A, msgId_4, C, \{[I, role, org, \{u_{id}\}_{K_B^+}, \{u_{id}\}_{K_C^+}]\}_{K_I^-}, XCAData & (4) \\
C \rightarrow I & : & C, msgId_4, I, \{[I, role, org, \{u_{id}\}_{K_B^+}, \{u_{id}\}_{K_C^+}]\}_{K_I^-}, RST(remgw) & (5) \\
I \rightarrow C & : & I, msgId_4, msgId_5, C, RSTR(\{[I, role, org, \{u_{id}\}_{K_{remgw}^+}]\}_{K_I^-}) & (6)
\end{array}
$$

Table 1: The delegate assertion usage

The notation, commonly used to describe security protocols is as follows. $\{M\}_{K_A^+}$ stands for the encryption of message M using the public key of A and $\{[M]\}_{K_A^-}$ for the signature of M using As private key (where $[M]$ is the hash code of M). $ts, ts, ts1$ and $ts2$ are timestamps.

In step 1, the consumer named A requests a SAML assertion with the local username *user* to the identity provider I by passing its name, A, a message identifier $msgId_1$ and the uri I, that represents the WS-Addressing `From`, `To`, `MessageID`. The SOAP Body of the message contains a signed request security token (RST) where the client application sends the requested user, role and organization. The last two values, B and C are the endpoints of the registry and the initiating gateway where the assertion will be used. The identity provider validates the message and issues the SAML assertion, by encrypting the value of the new SAML username u_{id} for the endpoints requested and adds a new message identifier ($msgId_2$) and the old message identifier, as `RelatesTo` WS-Addressing element. At this stage (step 3) the consumer A uses the obtained SAML assertion for contacting the registry (B) and the initiating gateway (C). Both services are able to decrypt the value of the username. The representation of the assertion is the signed tuple is $\{[I, role, org, \{u_{id}\}_{K_B^+}, \{u_{id}\}_{K_C^+}]\}_{K_I^-}$. We provide a formal account of this motivating example in [14].

3.1 The delegate assertion

In our model, the layout of the identity assertion is given by [19, 5]. The assertion contains the *role*, the *institution*, the *username* and the *purpose of use* of the requester (i.e. the doctor). In order for the remote site to send audit trails with the correct subject and to use attributes to enforce access

can be found in the full version [14].

control policies, it is important to attest the digital identity including the attributes of the user that is requesting health data.

As we can derive from the counterexample shown in Table 1, the responding gateway of the foreign community does not share the public keys (K_B, K_C). Thus it can't decrypt the `EncryptedID` value of the SAML assertion that is forwarded by the initiating gateway and therefore a new assertion is needed.

In [13] a list of trust models are presented. The model that we use is the *direct brokered trust*: the responding gateway of the foreign community trusts the initiating gateway of the local community that vouches for the document consumer of the local affinity domain.

When the document consumer needs to perform a cross gateway query, it contacts the initiating gateway with the SAML assertion containing the local subject. In order to establish the direct brokered trust in cross community transactions, the initiating gateway needs to vouch for the document consumer with the responding gateway in the foreign community.

Since the attributes as defined in [19] for attesting the digital identity of the requesting subject (i.e. the doctor sitting on the document consumer) are defined in the local authentication assertion, the initiating gateway requests a new assertion, using the local SAML authentication assertion as authentication statements against the identity provider. In this way, the identity provider validates the local SAML assertion and issues a new token containing the original attributes such as role and organization. The subject of the assertion becomes the endpoint of the initiating gateway.

This behavior is shown in steps 5 and 6 of Table 1. The initiating gateway C sends to the identity provider a request security token using the assertion as authentication statement and the address of the remote gateway as list of service where it intends to use the assertion. The identity provider validates the request, validates the assertion and issues the delegate assertion used by C for any cross community transaction (step 6).

In order to establish the brokered trust, we adopt the SAML delegation mechanism [20]. The `Conditions` element of the new delegate assertion is enriched with the `DelegateRestrictionType`, containing two values: the instant when the identity provider delegated the initiating gateway to act on behalf of the user and the subject of the assertion. The SAML condition element becomes

```
<saml:Condition
  xmlns:del="urn:oasis:names:tc:SAML:2.0:conditions:delegation"
  xsi:type="del:DelegationRestrictionType">
  <del:Delegate
```

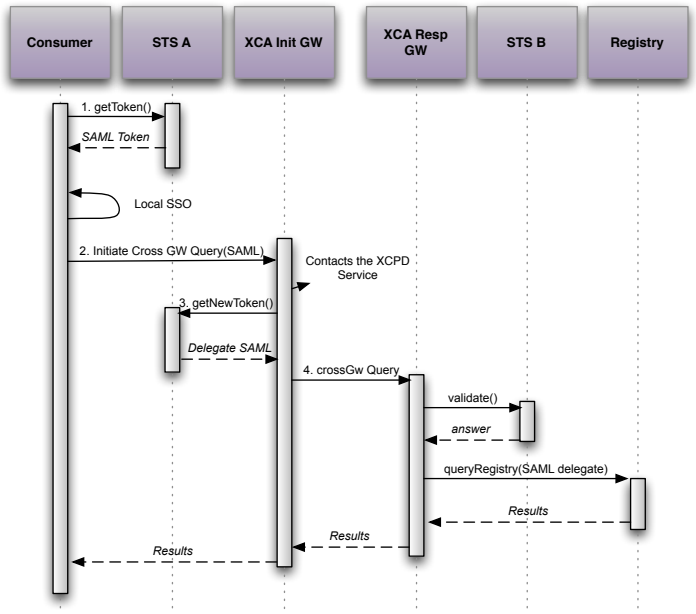

Figure 3: Applying security

```
  ConfirmationMethod="urn:oasis:names:tc:SAML:1.0:cm:sender-vouches"
  DelegationInstant="2010-05-13T12:50:30.846Z">
    <saml:NameID>drmasi@hospital.at</saml:NameID>
  </del:Delegate>
</saml:Condition>
```

The UML sequence diagram of the new scenario is depicted in Figure 3. The `AudienceRestriction` element of the delegate assertion now contains the list of the web service endpoints of the known gateways of the other communities and the topmost XCPD service. The patient identifier discovery transaction is also authenticated using the delegate assertion. Notably if there is no a topmost XCPD service, the `AudienceRestriction` list shall contain all the web services endpoint of all the XCPD services in all communities.

We used our solution in the e-Care pilot project in an Austrian region (presented at the IHE Connectathon 2008 [21]). In this pilot three hospitals need to cooperate together to exchange the EHR of the patients. The hospitals are identified as communities. A requirement of the project is to enforce at the remote site a patient consent (i.e. a policy) that the patient gives at the hospital where she is receiving treatment. For this reason when the initiating gateway performs a cross hospital transaction, it obtains the SAML delegate assertion and another treatment-related assertion that contains the encrypted local policy. Initiating gateway uses the delegated assertion for performing the patient discovery transaction and when the patient is found (and the community containing data is found) initiating gateway

sends the delegate assertion and the treatment assertion containing the policy encrypted for the responding gateway. The access control methodology is XACML-based and the access request is built upon the attributes of the delegated assertion and on the document metadata (the *resource*). The responding gateway grabs the policy from the encrypted assertion and perform a XACML flow, by importing also local policies.

4 Conclusions

We have presented our experience on the application of the SAML delegate assertion in an healthcare scenario as defined by the IHE standards. Specifically we have considered a scenario including a number of communities, a set of healthcare facilities cooperating together exchanging data over gateways. We proposed the usage of the OASIS brokered trust scenario where the initiating gateway obtains a new authentication assertion on behalf of the real user operating in the local community. We provided a methodology for the exchange of the user attributes over cross community transactions in order to give to the remote gateway the possibility to enforce local or remote access control policies.

Our methodology is well suited where the access control methodology is made using a XACML [22] flow, where the access request's subject is obtained from the attributes of the authentication assertion sent along the transaction.

We implemented our proposal using web service technologies such as WS-Security [17], WS-Trust [11], SAML profile for XACML [23]. A prototypal version of our software is running in a pilot project connecting three hospitals (communities) in a Austrian region and it will be extended to a governmental level in Austria. An STS capable of issuing the assertion with the proposed layout is available at `http://office.tiani-spirit.com:41081/SpiritIdentityProvider/listServices`. We have provided a full version of this work that includes the layout of the delegated assertion and a formalization of the motivating example. More information can be found in [14]. Our solution is at the time of writing under discussion as default methodology for propagating security claims on the eHR project in South Africa [3].

Related and Future works: There are no known relevant work related to the usage of SAML delegate assertions. In the european project epSOS [2], the information about the local subject is propagated by placing it in the attribute of the form [19] which does not follow the emerging SAML standard [20]. The Liberty Alliance Project [24] proposes a set of specifications that covers the problem of brokering trust. However these specifications are

not selected by IHE. The access control methodologies are defined by the IHE White Paper on Access Control [25]. The usage of XACML in conjunction with attributes [19] is recommended, as in our case.

To simply adopt WS-Security, WS-Trust and SAML does not guarantee absence of security flaws. Due to the complexity of the healthcare applications we plan in the near future to provide a formal methods-based analysis in order to investigate on the absence of flaws such as rewrite or protocol attack.

References

[1] ARGE-ELGA: Die österreich elektronische gesundheitsakte (2008) http://www.arge-elga.at.

[2] The epSOS project: a european ehealth project (2010) `http://www.epsos.eu`.

[3] The South African Department of Health: the EHR project in south africa (2009) `http://southafrica.usembassy.gov/root/pdfs/pepfar-hmis-docs/ndoh-e-hr-for-south-africa.pdf`.

[4] GIP DMP: Dossier Médical Personnel (2009) A French Project `http://www.d-m-p.org`.

[5] The Nationwide Health Information Network (NHIN): an American eHealth Project (2009) `http://healthit.hhs.gov/portal/server.pt`.

[6] The IHE Initiative: IT Infrastructure Technical Framework (2009) `http://www.ihe.net`.

[7] Health Level Seven organization: Hl7 standards (2009) `http://www.hl7.org`.

[8] ACR-NEMA: Digital Imaging and Communications in Medicine (DICOM) (1995)

[9] 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`.

[10] OASIS/ebXML Registry TC: ebXML business process specification schema technical specification v2.0.4. (2006) `http://www.ebxml.org`.

[11] 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.

[12] Witting, K.: Cross Community Access profile (2008) http://wiki.ihe.net.

[13] OASIS Web Services Security TC: Trust Models Guidelines. (2004)

[14] Masi, M., Maurer, R.: On the usage of SAML delegate assertions in an healthcare scenario with federated communities (full version). Technical report, Dipartimento di Sistemi e Informatica, Univ. Firenze - Tiani Spirit, Wien (2010) Available at http://rap.dsi.unifi.it/cows.

[15] Masi, M., Meoni, M.: Using Integrating the Healthcare Enterprise (IHE) profiles for an healthcare DataGRID based on AliEn. In: Emmit, AITIM (2008)

[16] IHE Technical Committee: Cross Community Patient Discovery (2009) http://www.ihe.net/Technical_Framework/upload/IHE_ITI_TF_Supplement_XCPD_PC_2009-08-10.pdf.

[17] OASIS Web Services Security TC: Web service security: SOAP message security (2006) http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf.

[18] Masi, M., Pugliese, R., Tiezzi, F.: On Secure Implementation of an IHE XUA-Based Protocol for Authenticating Healthcare Professionals. In: ICISS. Volume 5905 of LNCS., Springer (2009) 55–70

[19] OASIS eXtensible Access Control Markup Language TC: Cross Enterprise Security and Privacy Authorization Profile for XACML for healthcare. (2009) http://docs.oasis-open.org/xacml/xspa/v1.0/xacml-xspa-1.0-os.html.

[20] OASIS Web Services Security TC: SAML V2.0 Condition for Delegation Restriction Version 1.0 (2009) http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-delegation.html.

[21] Aichinger, B.: e-Care project presentation (2008) http://www.ihe-austria.at/fileadmin/user_upload/CAT2009/documents/IHE_CAT09_WSProgram_englisch.pdf.

[22] eXtensible Access Control Markup Language TC v2.0 (XACML): Extensible access control markup language (XACML) version 2.0 (2005) http://docs.oasis-open.org/xacml/2.0/XACML-2.0-OS-NORMATIVE.zip.

[23] eXtensible Access Control Markup Language TC v2.0 (XACML): SAML 2.0 profile of XACML v2.0 (2005)

[24] The Liberty Alliance: Project Liberty (2010) `http://www.projectliberty.org/`.

[25] The IHE Initiative: IHE Access Control White Paper. (2009) `http://www.ihe.net/Technical_Framework/upload/IHE_ITI_TF_WhitePaper_AccessControl_2009-09-28.pdf`.

[26] Armando et al, A.: 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)

[27] Johnson, J., Langworthy, D., Lamport, L., Vogt, F.: Formal specification of a web services protocol. Journal of Logic and Algebraic Programming **70**(1) (2007) 34 – 52

[28] Bhargavan, K., Corin, R., Fournet, C., Gordon, A.: Secure sessions for web services. In: SWS, ACM (2004) 56–66

[29] Abadi, M., Gordon, A.: A calculus for cryptographic protocols: The spi calculus. Inf. Comput. **148**(1) (1999) 1–70

[30] Bhargavan, K., Fournet, C., Gordon, A., Pucella, R.: TulaFale: A Security Tool for Web Services. CoRR **abs/cs/0412044** (2004)

[31] Lapadula, A., Pugliese, R., Tiezzi, F.: A Calculus for Orchestration of Web Services. In: ESOP. Volume 4421 of LNCS., Springer (2007) 33–47

A SAML Assertion layout

In this section we report the full encoding (without obvious namespace declarations) of the delegate assertion issued by our implementation running in the pilot project in Austria [21].

Listing 1: SAML Delegate Assertion

```
<saml:Assertion
 xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
 ID="_105b8d8b75f9edbca3ce5dba71a0b2bf"
 IssueInstant="2010-05-13T12:50:30.846Z" Version="2.0">
  <saml:Issuer Format="#nameid-format:entity"
   NameQualifier="urn:oo-project:issuer:namequalifier"
   SPNameQualifier="urn:oo-project:issuer:namequalifier">
   http://localhost:8081/SpiritIdentityProvider/services/
      SpiritIdentityProvider
  </saml:Issuer>
  <ds:Signature>
   <ds:SignedInfo>
    <ds:CanonicalizationMethod
     Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
     <ds:SignatureMethod
      Algorithm="#rsa-sha1"/>
     <ds:Reference URI="#_105b8d8b75f9edbca3ce5dba71a0b2bf">
      <ds:Transforms>
       <ds:Transform
           Algorithm="#enveloped-signature"/>
       ...
      </ds:Transforms>
     </ds:Reference>
    </ds:SignedInfo>
  </ds:Signature>
  <saml:Subject>
   <saml:NameID Format="#emailAddress">
    urn:initiating-gw-gespag-cc
   </saml:NameID>
   <saml:SubjectConfirmation
    Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
    <saml:SubjectConfirmationData>
     <ds:KeyInfo>
      <ds:X509Data>
       <ds:X509Certificate>MIII...</ds:X509Certificate>
       <ds:X509SubjectName>CN=EHR_SPIRIT...
       </ds:X509SubjectName>
       <ds:X509IssuerSerial>
        <ds:X509IssuerName>EMAILADDRESS=eric.poiseau@inria.fr
        </ds:X509IssuerName>
        <ds:X509SerialNumber>70</ds:X509SerialNumber>
```

```
      </ds:X509IssuerSerial>
     </ds:X509Data>
    </ds:KeyInfo>
   </saml:SubjectConfirmationData>
  </saml:SubjectConfirmation>
  </saml:Subject>
  <saml:Conditions
   NotBefore="2010-05-13T12:50:29.525Z"
   NotOnOrAfter="2010-05-13T13:00:29.522Z">
   <saml:AudienceRestriction>
    <saml:Audience>http://www.gespag.at/XCA/gateway
    </saml:Audience>
    <saml:Audience>http://www.kwg.at/XCA/gateway
    </saml:Audience>
   </saml:AudienceRestriction>
   <saml:Condition
    xmlns:del="#conditions:delegation"
    xsi:type="del:DelegationRestrictionType">
    <del:Delegate
     ConfirmationMethod="#sender-vouches"
     DelegationInstant="2010-05-13T12:50:30.846Z">
     <saml:NameID
      Format="#emailAddress">masi@gespag.at</saml:NameID>
    </del:Delegate>
   </saml:Condition>
  </saml:Conditions>
  <saml:AuthnStatement
   AuthnInstant="2010-05-13T12:50:30.820Z"
   SessionNotOnOrAfter="2010-05-13T12:50:30.846Z">
   <saml:AuthnContext>
    <saml:AuthnContextClassRef>
     urn:oasis:names:tc:SAML:2.0:ac:classes:PreviousSession
    </saml:AuthnContextClassRef>
   </saml:AuthnContext>
  </saml:AuthnStatement>
  <saml:AttributeStatement>
   <saml:Attribute FriendlyName="XSPA
    subject" Name="#xacml:1.0:subject:subject-id"
    NameFormat="#attrname-format:uri">
    <saml:AttributeValue
     xsi:type="xs:string">masi@gespag.at
    </saml:AttributeValue>
   </saml:Attribute>
   <saml:Attribute
    Name="#subject:organization">
    <saml:AttributeValue>Gespag</saml:AttributeValue>
   </saml:Attribute>
   <saml:Attribute
    Name="#subject:hl7:permission">
```

```
90       <saml:AttributeValue>hl7:POE-006</saml:AttributeValue>
91      </saml:Attribute>
92      <saml:Attribute Name="#2.0:subject:role">
93       <saml:AttributeValue>Dentist</saml:AttributeValue>
94      </saml:Attribute>
95      <saml:Attribute Name="#subject:purposeofuse">
96       <saml:AttributeValue>TREATMENT</saml:AttributeValue>
97      </saml:Attribute>
98     </saml:AttributeStatement>
99 </saml:Assertion>
```

B A formal account of the considered setting

Due to the increasing complexity of healthcare related projects, risk assessments related to security issues and possible flaws shall be done. The messages involved in a cross community transaction are numerous and prone to security flaws. It is sometime unfeasible to perform a risk analysis by means of standard techniques: the usage of formal methods is required in most of the cases. It is worth noticing that these techniques obtained interesting results on the discovering of flaws [26, 18, 27, 28]. For this reason we here give a formal account of the setting described in Table 1 to show the feasibility of this line of research. We leave open to future enhancements the analysis of properties of the formal model described here.

For the formalization of the motivating setting described in Section 3 we exploit the syntax of the Spi-Calculus [29] because it has in its semantic constructs for handling tuples such as digital signature, public key encryption and symmetric key cryptography. We are not interested here in dealing with more complex calculi such as [30] since the layout of our messages are fixed by the standards thus we can concentrate just on the value passed.

The Spi-Calculus is a small but extremely expressive programming language. Programs are systems independent, parallel processes that synchronize via message passing handshakes on named communication channels. These channels may be *restricted*, so that only certain processes may communicate on them.

We review a subset of the syntax of the Spi-Calculus in order to ease the reading of the model. We refer the interested reader to a complete description of the calculus with properties and proofs to [29].

We assume an infinite set of *names* to be used for communication channels, and an infinite set of *variables*. We let m, n, p ,q, and r range over names and let x, y, and z range over variables. The syntax of terms and processes are respectively shown in Table 2 and Table 3.

$L, M, N ::=$	terms
n	name
(M, N)	pair
0	zero
$suc(N)$	successor
x	variable
$\{M\}_N$	shared-key encryption
M^+	public part of a key
M^-	private part of a key
$\{[M]\}_N$	public-key encryption
$[\{M\}]_N$	private-key signature

Table 2: Syntax of terms in Spi-Calculus

$P, Q, R ::=$	processes
$\overline{M}\langle N\rangle.P$	output
$M(x).P$	input
$P\|Q$	composition
$(\nu n)P$	restriction
$!P$	replication
$[M \ is \ N] \ P$	match
$\mathbf{0}$	nil
$let \ (x, y) = M \ in \ P$	pair splitting
$case \ M \ of \ 0 : P \ suc(x) : Q$	integer case
$case \ L \ of \ x_N \ in \ P$	shared-key decryption
$case \ L \ of \ \{[x]\}_N \ in \ P$	public-key decryption
$case \ N \ of \ [\{x\}]_M \ in \ P$	signature check

Table 3: Syntax of processes in Spi-Calculus

We give here an intuitive account of the syntax of the processes of the Spi-Calculus. The basic synchronization primitives are *input* and *output*. An *output process* $\overline{M}\langle N\rangle.P$ is ready to output the message N on channel M, then P runs. An input process $M(x).P$ is ready to input on channel M. If an interaction occurs, term in which N is communicated on M, then the process P runs, where all the occurrences of the variable x are substituted by the value N. A composition $P|Q$ behaves as processes P and Q running in parallel that may interact. A restriction $(\nu n)P$ creates a new name n and then behaves as P. A replication $!P$ behaves as infinite copies of P running in parallel. A match, $[M\ is\ N]\ P$ runs as P provided that M is equal to N, otherwise the process is stuck. The nil process $\mathbf{0}$ does nothing. A pair splitting $let\ (x, y) = M\ in\ P$ behaves as P where all the occurrences of N, L are substituted by x, y if the term $M = (N, L)$, otherwise the process is stuck. The integer case, $case\ M\ of\ 0 :\ P\ suc(x)\ :\ Q$ behaves as P if the term M is 0, as Q with x replaced by N if M is $sucN$. Otherwise the process is stuck.

The encryption and decryption primitives are similar: the process attempts to encrypt/decrypt the message using the corresponding shared or public/private key and if succeeds behaves as P, otherwise the process is stuck. The same applies to the signature check: $case\ N\ of\ [\{x\}]_M\ in\ P$ means that if the process is able to validate the signature of the received message N, then behaves as P with the variables x substituted by N, otherwise the process is stuck.

The values effectively exchanged amongst the consumer A, the identity provider I and the initiating gateway C are shown in equation 1.

We abstract from the protocol used for issuing the token because it is out of scope.

The flow shown in equation 1 begins when A asks the identity provider for a new assertion making an output to the public channel c_{AI}. In the request, it places the username, the role, the organization and the services that the document consumer needs to contact, registry and initiating gateway. At this point the identity provider inputs from the channel and checks if the signature of the request is correct. If it is not correct the process is stuck. Then the identity provider creates a new SAML assertion and returns it to the document consumer.

The consumer now uses this assertions by making two parallel outputs: one to the registry B[2] and the other to the initiating gateway C. Notably, the private name *xds* is used to abstract the XDS registry details, and the name *xca* is inside the scope of *xds* to show that it "inherits" the data from the XDS query, in some way (further details are shown in [6]).

[2]The code for the registry is not shown here for the sake of simplicity

When contacted, C validates the SAML assertion (with the hidden function $I(x)$ not shown) and it requests the new delegated assertion by making output to the channel c_{CI}. It creates firstly the name of the remote gateway (that is in the real world the IP address) and forwards the SAML assertion received. The identity provider validates the assertion by checking the signature (other actions are not shown) and outputs to the channel $\overline{c_{IC}}$ the delegated assertion that is abstracted by placing the x_1 value in the fourth position of the tuple and by encrypting the name of the responding remote gateway. Then the initiating gateway performs the cross gateway query abstracted by the function $G()$.

It is worth noticing that each message exchanged represents a SOAP envelope and in the Spi-Calculus definitions there is no knowledge of SOAP header and SOAP body.

$$
\begin{aligned}
A(user, role, org) \triangleq\ & \overline{c_{AI}}\langle user, role, org, reg, initgw, \\
& \quad [\{H(user, role, org, reg, initgw)\}]_{K_A^-}\rangle. \\
& c_{IA}(x_1, x_2, x_3, x_4, x_5, x_6).(\nu xds) \\
& \Big(\overline{c_{AB}}\langle x_1, x_2, x_3, x_4, x_5, x_6, xds\rangle \mid \\
& \quad (\nu xca)\overline{c_{AC}}\langle x_1, x_2, x_3, x_4, x_5, x_6, xca\rangle\Big).0 \\
I \triangleq\ & \Big(c_{AI}(x_1, x_2, x_3, x_4, x_5, x_6).case\ x_6\ of\ [\{z\}]_{K_A^+}\ in \\
& \quad [H(x_1, x_2, x_3, x_4, x_5)\ is\ z]. \\
& \quad (\nu u_1)\overline{c_{IA}}\langle x_1, x_2, x_3, \{u_1\}_{K^+_{x_4}}, \{u_1\}_{K^+_{x_5}}, \\
& \quad\quad [\{H(x_1, x_2, x_3, \{u_1\}_{K^+_{x_4}}, \{u_1\}_{K^+_{x_5}})\}]_{K_I^-}\rangle\Big) \\
& \Big|\Big(c_{CI}(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8). \\
& \quad case\ x_6\ of\ [\{z\}]_{K_I^+}\ in\ [H(x_1, x_2, x_3, x_4, x_5)\ is\ z]. \\
& \quad \overline{c_{IC}}\langle x_2, x_3, \{x_4\}_{K^+_{x_8}}, x_1, \\
& \quad\quad [\{H(x_2, x_3, \{x_4\}_{K^+_{x_8}}, x_1)\}]_{K_I^-}\rangle\Big) \\
C \triangleq\ & c_{AC}(x_1, x_2, x_3, x_4, x_5, x_6, xca).I(\mathbf{x}).(\nu remgw) \\
& \quad \overline{c_{CI}}\langle x_1, x_2, x_3, x_4, x_5, x_6, C, remgw\rangle. \\
& \quad c_{IC}(x_2, x_3, y_1, x_1, y_2).G(x_2, x_3, y_1, x_1, y_2, xca)
\end{aligned}
\tag{1}
$$

A community is seen as the parallel composition of the actors of equation 1. We assume a pre-deployment of private and public keys[3].

$$Community_A \triangleq (\nu initgw)(\nu remgw)(\nu K_A)(\nu K_I)(\nu K_C)(\nu K_{remgw}) \left((\nu user)(\nu role)(\nu org)(\nu reg)\ !A(user, role, org) | !I | C \right) \quad (2)$$

Other communities are added as parallell processes to the system

$$Sys \triangleq Community_A \mid Community_B \quad (3)$$

As we can see from the example setting, it is relatively straightforward to encode a protocol informally written as in Table 1 to a process calculus such as the Spi-Calculus. Using this technique, security properties can be verified by means of bisimulation or model checking, by also relying on more ad hoc calculi such as COWS [31], as seen in [18].

[3]The key exchange process is not shown, for the sake of simplicity