Sindh Univ. Res. Jour. (Sci. Ser.) Vol.44 (3) 479-486 (2012)

SINDH UNIVERSITY RESEARCH JOURNAL (SCIENCE SERIES)

Implementing SOA Security Using (SCA) Service Component Architecture, Model

M. MEMON, G.D. MENGHWAR, M. H. DEPAR. A. A. JALBANI
(mukhtiar.memon, gdas, mansoor.hyder, akjalbani)@sau.edu.pk

Information Technology Center, Sindh Agriculture University, Tandojam, Pakistan

Received 24th April 2012 and Revised 12st August 2012

Abstract: Security can be implemented for service-oriented systems using a variety of security architectural models. Conventional security architectures rely on endpoint security having hard-coded and tightly-coupled security and business procedures. Due to this it is not possible to separate business and security protocols. As a result, the service endpoints have to execute both the security and business protocols creating performance problems for the business services. Besides, it becomes difficult to develop, maintain and update the security components independently due to tight integration of business and security code. To solve this problem it is important to separate the business and security concerns in SOA systems and provide security by dedicated security services offered by decoupled security components. This paper presents a SOA security solution based on service-oriented security architecture to execute out-of-band security protocols. The proposed approach is implemented with Service Component Architecture (SCA), which provides an architectural model for composition of security services in SOA systems. The proposed approach is applied for the security requirements of healthcare systems to protect the sensitive medical data of the patients from unauthorized exposure.

Keywords: Security service-oriented systems models

1. INTRODUCTION

Securing SOA-based system poses many challenges to security architects. Conventional endpoint security achieves only primitive security requirements for physically scattered and logically composed SOA systems. If security is coupled with the service endpoints then all the processing required for security implementations is carried out by the service endpoints, which increases the burden on the service side (Kanneganti and Chodavarapu, 2007). Moreover, the different service endpoints use different security infrastructures supporting variety of security mechanisms for authentication, authorization etc., therefore service invocation has to be compatible to those security infrastructures. This paper presents an alternative solution for SOA security based on service-oriented security approach. The main motivation of this work is to decouple security functionality from functional (business) services. There are many advantages for this separation. For instance, if security is dealt separately then many services can benefit from its functionality, because the same service is shared by a number of service endpoints. Besides, it becomes easy to deploy, configure, control and maintain the security infrastructures without affecting the business services. Performance is one of the critical issues in service-oriented systems. If security functionality is offered by dedicated security services then the service endpoints can focus on functional aspects of the services. The security aspects can be addressed by security service provider e.g., security broker, trusted third party, etc. As a major additional advantage, service-oriented security also provides the reusability by sharing the security functionality. With shared security the heterogeneity of the security data (tokens, policies, certificates, assertions etc.) can be resolved by shared security services, which otherwise would have been the responsibility of service endpoints. Furthermore, monitoring for quality of service parameters such as performance and availability will be easier and effective in service-oriented security. The performance of the business services will increase, because the computation-intensive security tasks including execution of security protocols for authentication, authorization, non-repudiation and monitoring will be performed by the dedicated security components. We have proposed to implement the service-oriented security using Service Component Service (SCA), which provides a technology model to compose services from components (Satoh, Nakamura and Ono, 2006). Using this model the security components are composed to meet certain requirements such as authentication, authorization, privacy and logging (Memon, Hafner and Breu, 2008).

2. MATERIALS AND METHODS

Contributions:

The proposed approach contributes couple of interesting and innovative ideas. Primarily, our approach provides a conceptual foundation of separation of security and business concerns to achieve better

++ Corresponding author: M. MEMON mukhtiar.memon, gdas.

performance, maintainability and componentization properties in the SOA security architectures. Secondly, the proposed security composition architecture is underpinned by the SCA composition model, which is very useful to compose systems out of distributed components. In the present status the SOA community has only focused composition of business components or services. This is the only kind of work based on SCA model to compose security components to achieve SOA security. Further contribution is in the direction of security requirements for healthcare systems, which process sensitive medical data of patients in multi-party healthcare services. We have identified the basic as well as advanced security requirements for healthcare systems and proposed a SOA security solution for protecting the medical data. On the technology side, the proposed reference architecture uses contemporary web services security standards such as SAML, XACML, WS-Trust etc. for most up-to-date industry-based deployment of security services at the SOA security architecture.

The remaining paper is organized as follows. Section 2 presents related research and approaches. Security requirements for motivating example of healthcare systems are discussed in Section 3. Proposed Service-oriented security architecture is presented in Section 4. Section 5 complements the proposed reference architecture with implementation using Service Component Architecture (SCA) model for complex security requirements of healthcare systems. We conclude the discussion in Section 6.

3. Related Work

(Wolter, Menzel and Meinel, 2008) have discussed security for service composition and cross-organization service invocation in SOA systems. However, they have not mentioned where and how security functionality is achieved because they have focused on the composition of business services. On the contrary, our solution provides the architecture for SCA-based composition of security services, which are solely dedicated to provide security functionality of authentication, authorization, privacy and non-repudiation. The Security Modeling approach presented by (Basin, Doser and Lodderstedt, 2006) defines security with functional model using RBAC (Ferraiolo, Sandhu, Gavrila, Kuhn and Chandramouli, 2001) permissions and code-generation for security policies. We use commonly practiced security patterns for high-level security modeling as discussed by (Cheng, Konrad,. Campbell and Wssermann, 2003) and (Rosado, Gutierrez, Fernandez-Medina and Piattini, 2006). These approaches have merely focused the high-level design of the secure SOA systems. Whereas, we constitute the concrete parts of SOA security i.e., security architecture and security components.

Security-as-a-Service approach for SOA systems is presented by (Hondo, Hinton and Hutchison, 2005). Antivirus services use service-oriented security for small and medium businesses (MacAfee White Paper, 2008). A very comprehensive discussion about service-oriented security is provided by (R. Kanneganti and P. Chodavarapu, 2007), applying the approach for authentication, trust and secure conversation. The main motivation of using service-orientation for security given is the decoupling of security processing from services. The authors suggest shifting the burden of security implementation from service endpoints to shared security services. These approaches are in the same direction as our research; however, none of them have used compositional aspects of security components, which is our novel idea.

The focus of this paper is to use service-oriented security for broader security objectives targeting complex security services such as Cross-domain Authentication, Distributed Authorization, Privacy and Anonymization, Auditing and Policy Validation. A Similar approach is presented by (Jose Vivas, Eiji Okamoto, Javier Lopez, *et. al.*, 2005), which also invokes separate security services. Security services in proposed work are composed from security components and composition is based on *Service Component Architecture* (SCA) model (OSOA - Service Component Architecture, 2007). SCA uses SCAPolicyFramework (Satoh,Nakamura, Mukhi, Tatsubori and Ono, 2008) to secure the functional components. However, the approach presented in this paper is different as the SCA model is applied for composing security services from *Security Components*.

4. Scenario: The Distributed Healthcare Systems Security

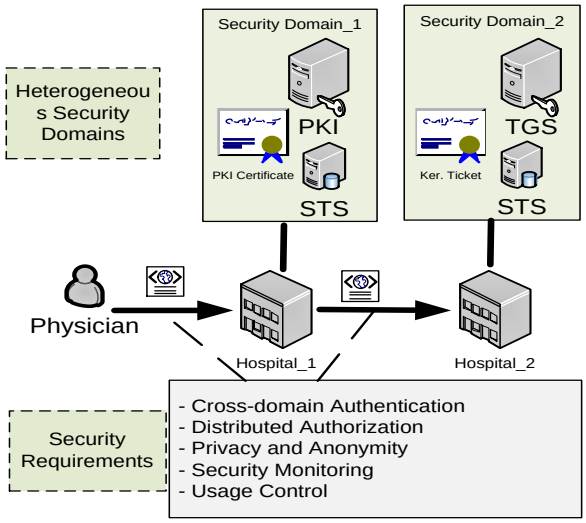


Fig. 1: Security Requirements in Service-oriented Healthcare System

Service-oriented healthcare systems are very complex, because they involve many organizations in the business workflows. For example, the services to access and process patient's medical record are distributed at different locations and involve cross-domain security interactions for authentication and authorization. **(Fig. 1)** shows business interactions among two healthcare organizations i.e., Hosptial_1 and Hosptial_2. The security infrastructures of those organizations are shown in their respective security domains i.e., Security Domain_1 and Security Domain_2, respectively. The security infrastructure in Security Domain_1 is based on Public Key Infrastructure (PKI), where Security Token Service (STS) provides security tokens for authentication and authorization. Similarly, Hospital_2 in Security Domain_2 uses Kerberos Ticket Granting Service (TGS) and corresponding STS for authentication and authorization. The authentication and authorization are required in both domains when a user i.e., Physician needs to access the medical records stored in these organizations. To grant access to the physician, the target endpoint has to communicate the security context of an authenticated requester to other service endpoints. This involves security processing for resolution of security tokens, policy decision, policy validation, privacy policy enforcement, anonymization and logging.

Fig. 1 also shows the complex security requirements required for secure interaction between the endpoints. Our previous work (Memon, Hafner, and Breu, 2008) presented the complex security requirements for healthcare scenarios in more detail. In this paper, it is proposed to solve such requirements using service-oriented security approach implementing *Service Component Architecture* (SCA) programming model, illustrating cross-domain authentication example. The same approach is also applied for more advanced security requirements in Section 5.2.

4 Service-Oriented Security Architecture

This section presents the particular problems related to SOA security and then discusses how service-oriented security can solve those problems. One of the options for securing SOA systems is by enforcing security at the service endpoints, which implies that the service provider is also responsible for securing the business (e.g., medical records) and security data (e.g., tokens, assertions, sessions, policies, etc.). On one hand, endpoint security has some advantages. For example, it would be easy to integrate, implement and manage the security controls. It has obvious advantages for primitive security requirements of encryption, signature etc., for efficiency and less risk, as the security processing is performed locally. On the other hand, it has many limitations in the perspective of securing SOA systems, in which business services and security controls can never be placed exclusively at one endpoint. For SOA systems, endpoint security poses following limitations; also discussed in (Kanneganti and Chodavarapu, 2007):

- All the endpoints have to deploy, configure and manage the security infrastructures internally, which is very costly and also increases security burden at the service provider's end.
- Due to heterogeneous security infrastructures at different service endpoints, it is difficult to achieve security interoperability.
- SOA security requirements are very complex and implemented in a variety of architectures and standards, which an endpoint cannot handle.
- The application developer at every endpoint has to consider a variety of security standards and come up with respective implementation solutions.

Moreover, in particular to service-oriented healthcare systems the security requirements are very complex and cannot be met by endpoint security. Some of the major security requirements are outlined below:

Cross-domain Authentication: The physician needs to authenticate other security domains with different security infrastructure to access the medical services.

Distributed Authorization: The physician gets authorization based on the policies at respective endpoints. The policies have to be negotiated by security services before granting permissions to the service requester.

Privacy: This is very demanding requirement and it is difficult to implement patients' privacy preferences and consents, when medical services retrieve the data from different locations (Yee, Korba and Song, 2006).

Anonymization: Medical records are also used for secondary purpose such as health research, surveys and marketing. The personal identification information should be anonymized with certain techniques such as pseudonymization before sensitive medical data is disseminated for secondary purpose.

Logging and Auditing: Access granted to a physician should be logged for future audit and accountability.

Policy Validation: As endpoint services have different policies to access the medical services. It is important to validate the policies, to check if those meet an endpoint's effective security policy.

It is very challenging to meet the above-mentioned security requirements of healthcare systems

with endpoint security due to its limitations discussed earlier in this section. The proposed service-oriented security approach can solve most of the above-mentioned problems for a variety of SOA application domains including healthcare. The service-oriented security approach can be defined as a security implementation for SOA systems, in which security services are provided as independent services and are not bound or dependent on the business service endpoints. An endpoint relies on the decision of shared security services and enforces the decision at its end. As most of the security requirements of different service providers are similar (i.e. authentication, authorization, privacy, non-repudiation, auditing and policy validation etc.), therefore a shared security service can practically provide services for them. The service provider is concerned only with the decision (as a policy or assertion), that results from the run of a security protocol. For instance, the decision that says if a physician is authenticated or if she has permissions to call a certain service. Similarly, for privacy, the concern of service provider is to retrieve the privacy policy that applies to the digital rights of patients, which could be remotely located and a shared service can provide that policy. For non-repudiation and auditing, the service provider is interested in logging the communication between requester and provider, which could be performed by security service as well. The service provider can later on perform auditing based on the logs for verifying the access violations or dispute resolution. Therefore, service-oriented security simplifies the service invocation to let the service provider focus on the functional aspects.

4.1 SAML-based Authentication

As discussed earlier, the service-oriented security approach can be used for many security requirements. For illustration purposes, the SAML-based authentication requirement is presented. It is based on an authentication policy defining the requirements of a service, which says that the service provider accepts the signed SAML tokens to grant access to the requested service. The example security policy for this scenario is shown below, which is one of the standard policies defined by OASIS in the WS-SecurityPolicy (OASIS - WS-SecurityPolicy Examples, 2007). The code for example policy is shown below. The first part of the policy uses signed supporting SAML tokens for authentication. The subsequent code shows that both the outgoing and incoming messages should be encrypted and signed for confidentiality and integrity.

```
<wsp:All>...
<sp:SignedSupportingTokens>
<wsp:Policy>
<sp:SamlToken
 sp:IncludeToken="AlwaysToRecipient">
<wsp:Policy><sp:WssSamlV11Token11/>
</wsp:Policy>
</sp:SamlToken>
</wsp:Policy>
 ...
</wsp:All>
...
<wsp:Policy wsu:Id="SamlForCertificates_input_policy">
<wsp:ExactlyOne><wsp:All>
<sp:SignedParts><sp:Body/></sp:SignedParts>
<sp:EncryptedParts> <sp:Body/></sp:EncryptedParts>
</wsp:All>
</wsp:ExactlyOne>
</wsp:Policy>
```

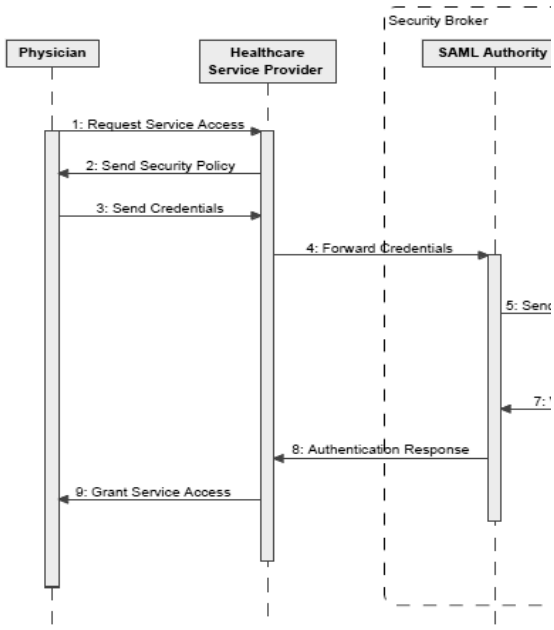


Fig. 2: SAML-based Authentication

```
<wsp:Policy wsu:Id="SamlForCertificate_output_policy">
<wsp:ExactlyOne><wsp:All>
 <sp:SignedParts> <sp:Body/>  </sp:SignedParts>
<sp:EncryptedParts>  <sp:Body/> </sp:EncryptedParts>
</wsp:All> </wsp:ExactlyOne>
 </wsp:Policy>
```

The implementation is done for the authentication protocol shown in Fig. 2. The protocol shows that the service provider calls services of a *Security Broker* for token validation, who is a security service provider. The *Security Broker* provides two services i.e., token validation by *Authenticator,* which relies on a *Security Token Service* (STS) and token assertion by a *SAML Authority*. The requester obtains the SAML authentication assertion from the *SAML Authority* and submits to the service provider as defined

in SAML Profiles (OASIS - Security Assertion Markup Language (SAML), 2005).

5. Implementing Sca Model For Service-Oriented Security

Section 4 presented the rational for using service-oriented security on the grounds that security requirements of a service endpoint can be handled by dedicated shared security services. The idea of a *Security Broker* was presented, who performs the necessary security-related interactions with the service requester and communicates the final decision to the service provider. This applies perfectly for authentication and authorization requirements in which the service provider needs to verify the service requester's identity claims and permissions assigned to him. The same approach could also be realized for privacy, anonymity, auditing, and policy validation as well. With the shared and composed security services, it becomes easy to develop, deploy, configure and manage security infrastructures needed for these requirements. Because, any updates in security-related APIs, mechanisms and algorithms etc. can be done without intervening the functional services. It also reduces the cost of implementation and management, because endpoints don't have to deploy and maintain the security infrastructures. On the basis of these facts the service-oriented security approach delivers a real implementation of technology- and platform-independent secure SOA systems.

The Service Component Architecture (SCA) provides the programming model to compose SOA systems from services offered by components in a platform-independent way (OSOA - Service Component Architecture, 2007). To benefit from this model, we modeled and implemented service-oriented security to meet the security requirements of SOA-based healthcare systems. The requirement of cross-domain authentication for healthcare systems is illustrated in more detail using SCA model. In the scenario, shown in Fig. 1, the security services are offered by *Security Components* and composed as a *Security Composite*, i.e., *Authentication Composite*, as shown in **(Fig. 3),** with a single interface to the set of security components.

There are many advantages of using component-based SCA composition model for service-oriented security approach for following reasons:

- The composition is dynamic and provides a common interface to the set of required security components. This feature of SCA can be used for implementing service-oriented security to compose the required components depending upon the service provider's policy. In that case, when a requester retrieves the policy it invokes security broker's shared security service to provide the functionality that meets the policy.
- The security services are not integrated with service endpoints, but operate independently. Therefore, any changes in the security functionality (e.g. security APIs, protocols etc.) can be done without affecting the service endpoints. This is one of the fundamental requirements of SOA security that an application should not depend on infrastructure of security provider.
- Policies are applied on components at their interfaces and references to apply constraints on the execution of services offered by them.
- The process of generating the definition of a Security Composite, which is written in Service Composition Definition Language (SCDL), is generated by applying mapping rules with the corresponding security policy. The example code in Section 5.1 shows the SCDL definition of a security composite for SAML-based authentication policy.

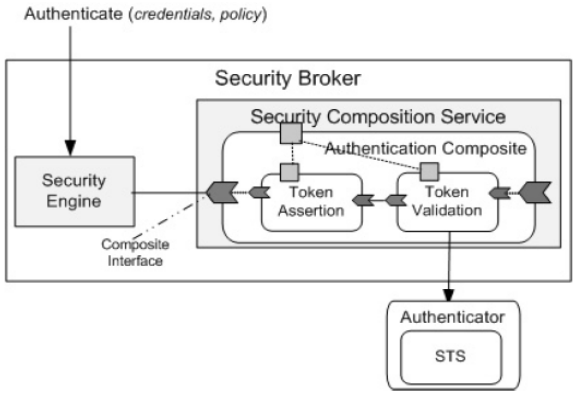


Fig. 3: Authentication Composite

5.1 Security Service Composition

One of the main motivations of using service-oriented security approach is to free service provider from burden regarding security implementation, processing and control. However, it is a challenging task to design architectures and select appropriate security standards that support this paradigm. Some of the standards, which can be implemented for service-oriented security, are SAML-based authentication, WS-Trust and WS-SecureConversation (R. Kanneganti and P. Chodavarapu, 2007). Based on these standards the SCA model can be used to solve cross-domain authentication.

The Fig. 3 shows an *Authentication Composite*. When a requester receives authentication policy from service provider, it sends the policy and credentials (signed and encrypted to be validated only by STS) to

the *Security Broker*. As, encryption, signature are primitive requirements, so we assume that the requester's platform is capable to perform these basic functions. The authentication policy of service provider defines its requirements, which requester should comply. For that purpose, it requests the authentication service of *Security Broker*. Requester sends the credentials and policy to security broker's *Security Engine*, which parses the policy and defines the set of security components which will be required to meet the policy requirements. As security policies are different based on requirements, therefore they require different implementations by security components. WS-SecurityPolicy scenarios defined in (OASIS - WS-SecurityPolicy Examples, 2007) presents number of standard authentication policies.

For SAML-based authentication policy, the SCDL code shown below defines an *Authentication Composite*, which composes two components i.e., *Token Assertion* and *Token Validation*. The security engine sends the policy parameters as properties to configure the implementation of the components with their values. Based on these values, the *Token Assertion* component is configured to generate a SAML token assertion (i.e., *WssSamlV11Token11*) and insert timestamp on it and use the assertion only once. The reference of this component shows that it uses the service of *Token Validation* component. The security engine configures the implementation of components for validation of *WssX509V3Token10* token and other requirements of policy such as encryption of body and header with *basic256* method, creation of message digest with *sha1* method and using *user name password* as supporting tokens.

```
<composite xmlns:sca=" . . /1.0"
 name="AuthenticationComposite">
. . . .
 <service name="AuthenticationService">
 <binding uri=".../authenticationService">

<component name="TokenAssertionComponent">
<implementation.java
 class="services.security.TokenAssertion"/>
. . .
<service name="tokenAssertion"/>
<property name="signedSupportedToken">
WssSamlV11Token11</property>
<property name="timestamp">true</property>
<property name="useonce">true</property>
<reference name="tokenValidationService"
target =" TokenValidationComponent"/>
</component>

<component name="TokenValidationComponent">
<implementation.bpel
process="services.security.tokenValidation"/>
<binding uri="services.security">
<service name="tokenValidationService"/>
<property name="bindingType">
asymmetric</property>
<property name="encrytionAlgorithm">
basic256</property>
<property name="digestMethod">sha1</property>
<property name="protectionToken">
WssX509V3Token10</property>
<property name="supportingToken"> Username</property>

<property name="encryptedParts"> Body</property>
<property name="encryptedParts">
Header</property>
</component></composite>
```

The security engine finds the required components and configures them with the external values (retrieved from policy) using the properties of the security component as shown in the code above. This is a very promising feature in SCA model to configure the implementation on-the-fly. It is very useful, because using this feature the requester can interoperate with a number of different security domains seamlessly. On behalf of requester this task is performed by the *Security Broker,* who is specialized in security and many service endpoints can share its services. For example, for healthcare services, the physician's local platform requests the trusted security broker for authentication and gets clearance for accessing distributed medical history of patient from different security domains. The same approach is applied for advanced requirements of healthcare systems discussed in subsequent part of this section.

5.2 Advanced Security Requirements

As discussed in Section 3, the healthcare scenarios have very complex requirements, but the service-oriented security can solve many of those requirements. The requirement of cross-domain authentication has been illustrated in Section 5.1. Similarly, security services composition can be performed for the advanced requirements such as *Authorization, Privacy*, *Anonymity*, *Auditing* and *Policy Validation*.

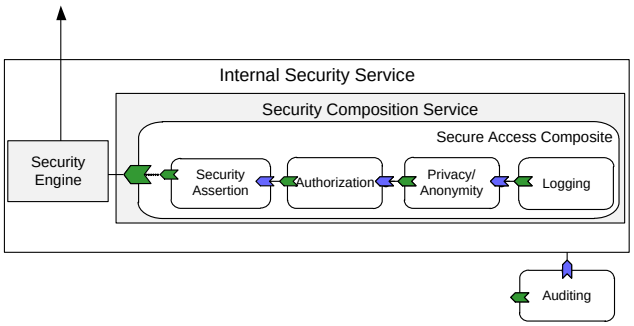


Fig. 4: Advanced Security Requirements

(Fig. 4) shows a *Secure Access Composite* for advanced healthcare security requirements. The Secure Access Composite is composed from security components to meet these requirements. The interface of the *Security Assertion* component is promoted as the interface of secure access component. The *Security Assertion Component* returns an authorization assertion to the security engine for the requested services. For the component of authorization the *Policy Decision Point* (PDP) is a shared security service which makes authorization decision and sends authorization assertion to the endpoint for enforcement.

However, before sending the authorization decision to the security engine the other components as *Privacy* and *Logging* play their roles. The privacy component retrieves patients' privacy policy and applies that policy on the medical records. It is why we have shown the authorization component calls the service of privacy component. Finally, the *Logging* component keeps track of the access granted by authorization component for future audit in the log database. The logged data stored by this component could be accessed by *Auditing* component for access verifications.

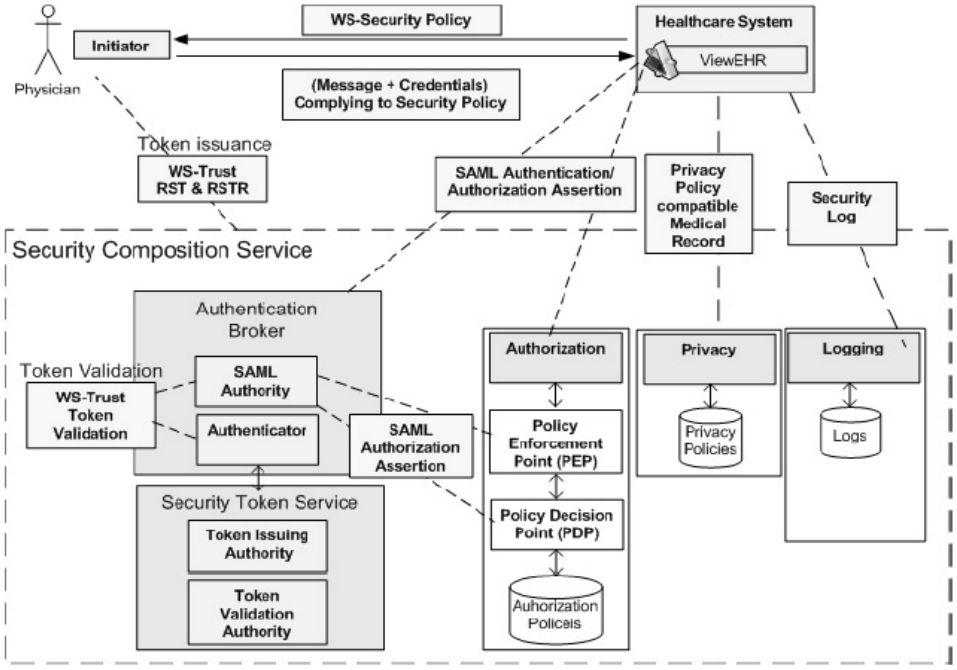


Fig. 5: The Reference Security Architecture

5.3 The Reference Security Architecture

Fig. 5 shows the reference architecture for implementation of security with service-oriented security approach using SCA model. It shows interfaces for communication among peers and the security artifacts used. The service provider (i.e. Healthcare System) sends its policy as WS-SecurityPolicy to the requester (i.e. initiator). The initiator interacts with security service on behalf of the physician. It sends security token to the *Authentication Broker* for validation and gets SAML authentication assertion, after the tokens are validated by *Authenticator*. Initiator uses WS-Trust interface to establish bootstrap trust with Authenticator.

It sends a *Request for Security Tokens* (RST) to the Authenticator who returns the tokens as *RST Response* (RSTR) as discussed in WS-Trust specifications (OASIS-WS-Trust Specifications, 2005). The service provider uses the WS-Trust interface such while validating the tokens of requester. Similarly, for authorization, the decision is made by *Policy Decision Point* (PDP) and sent to the *Policy Enforcement Point* (PEP) as *SAML Authorization Assertion*. For Privacy, the corresponding components apply rules defined in the privacy policies on the medical record. The *Logging* component creates the log of service access and stores in the log database for future audit by *Auditing* component.

6. CONCLUSIONS

In this paper, the service-oriented security implementation is proposed using Service Component Architecture (SCA). The limitations of endpoint security are discussed in the perspective of SOA security. The findings of this paper are that securing the

services at service provider's endpoint cannot meet SOA security challenges. The proposed approach based on shared security services on one hand reduces the security processing burden of service endpoint and on other hand it also resolves heterogeneity among security infrastructures at the service endpoints. SCA model is used to implement service-oriented security by composing security services from language- and platform- independent configurable security components. There are few issues related to real world implementation as security is tightly coupled in most of the service endpoints. Another big challenge is appropriate distribution of security processing between the service endpoints and security broker, because that depends upon many factors, such as application and security architecture, integration model and organizational policies.

REFERENCES:

Basin, D., J. Doser, and T. Lodderstedt (2006) Model Driven Security: From UML models to access control infrastructures. ACM Transactions Software Engineering Methodologies, 15 **(1):** 39–91.

Cheng, B.H.C., S. Konrad, L. A. Campbell and R. Wssermann (2003) Using security patterns to model and analyze security requirements. In IEEE Workshop on Requirements for High Assurance Systems, 13–22.

Ferraiolo, D. F., R. Sandhu, S. Gavrila, D. R. Kuhn and R. Chandramouli (2001) Proposed NIST standard for role-based access control. ACM Trans. Information Systems Security, 4 **(3):** 224–274.

Hondo M., H. Hinton and B. Hutchison (2005) Security Patterns within a Service-Oriented Architecture, http://www.ebizq.net/topics/woa/features/6535.html

Jose L. V. E. Okamoto, J. Lopez, J. A. Montenegro and Ed. Dawson (2005) Specification and Design of Advanced Authentication Authorization Services. Computer Standards and Interfaces, 27 **(5):** 467–478.

Kanneganti R. and P. Chodavarapu (2007). SOA Security in Action, Manning Publications Co., Greenwich, CT, USA.

MacAfee White Paper (2008) Security as a Service. http://www.mcafee.com/us/resources/white-papers/wp-security-opps-service-providers.pdf

Memon, M., M. Hafner, and R. Breu (2008). SECTISSIMO: A Platform-Independent Framework for Security Services. In ModSec '08: MODELS 2008, Toulouse, France.

OASIS. Security Assertion Markup Language (2005). http://www.oasis-open.org.

OASIS. WS-Trust Specifications (2005) http://docs.oasis-open.org/ws-sx /wstrust/ 200512/ws-trust-1.3-os.html.

OASIS. WS-SecurityPolicy Examples (2007) www.oasisopen. org/committees/download.php /23071/ws-sp-usecases-examples-draft-11-03.doc.

OSOA. Service Component Architecture (2007) http://www.osoa.org/display/Main/Home.

Rosado, D. G., C. Gutierrez, E. Fernandez-Medina and M. Piattini (2006) Security Patterns Related to Security Requirements. In WOSIS'06, 163–173.

Satoh, F., Y. Nakamura, N.K. Mukhi, M. Tatsubori, and K. Ono (2008). Methodology and Tools for End-to-End SOA Security Configurations. In Congress on Services - Part I, SERVICES '08, 307–314.

Satoh, F., Y. Nakamura, and K. Ono (2006). Adding Authentication to Model Driven Security. In ICWS '06: Proceedings of the IEEE International conference on Web Services, 585–594.

Wolter, C., M. Menzel, and C. Meinel (2008) Modeling security goals in business processes. In Modellierung '08, 197–212.

Yee, G., L. Korba, and R. Song (2006) Ensuring Privacy for E-Health Services. In ARES '06: Proceedings of the 1st International Conference on Availability, Reliability and Security, 321–328.