A Study on the Security Mechanism for Web Services

Kou Hongzhao

***Abstract*—Web Service has been widely used in the field of distributed application system. But the security issue of the Web Service has often been considered as a crucial barrier to its application in many fields that conducts sensitive information, such as E-commerce. In the paper, we introduce the Security Token Service (STS) into Web Service firstly, and then present a STS-based security architecture for Web Services and describe the architecture in some details.**

Index Terms—**Web Services, Security token, WS-*security specifications, *secure transactions*.**

I. Introduction

A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It is based on some common protocols, and the core of them are (1) Extensible Markup Language (XML), which include the Simple Object Access Protocol (SOAP), (2) the Web Services Description Language (WSDL), which is an interface described in a machine-processable format, and (3) Universal Description, Discovery, and Integration (UDDI), as Figure 1 shown.

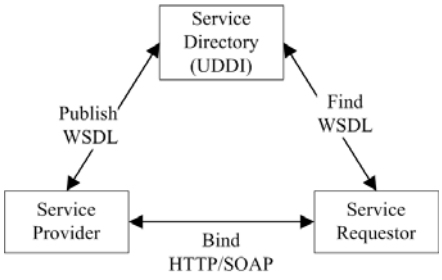

Figure 1 Web Service

All of these protocols are independent from the machine architecture, the underlying operating system and the programming language. Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards [1]. Using the Web service technology, the applications is allowed to communicate with each other in a platform- and programming language-independent manner.

A Web service provides a software interface that describes a collection of operations that can be accessed over the network through standardized XML messaging. It uses protocols based on the XML language to describe an operation to execute or data to exchange with another Web service. A group of Web services interacting together in this manner defines a particular Web service application in a Service-Oriented Architecture (SOA).

While Web Service is applied in system that conducts sensitive information, such as E-commerce, the security issues are considered. This means that Web Service needs to include features that can deal with security risks, including falsification and eavesdropping on data that is being transmitted over communication paths, as well as on spoofing and repudiation.

Before web service-related security standards became available, the Transport Layer Security (TLS) is a simply and widely used method for performing secure transactions for the Web security. But the TLS is aimed to authenticate the server hosting the Web Service, there is no means to authenticate a single service or sets of services running on the same machine. So it has many disadvantages in meet the security requirement for the Web Services system which are more complex than Web applications.

(i)TLS only provides point-to-point security, while a Web service messages often pass through multi- intermediaries before arriving the service providers, each intermediary may also make these arguments to the message, which means it is needed is end-to-end protection.

(ii) TLS provides security in the transport layer rather than in the message level. So the message is secure (encrypted) only in transmitting, and becomes plain-text when it at the end. An attacker can steal message from the message queue in the end.

(iii) TLS only provides an encrypted tunnel between the two communication parties by shared key established in handshakes. There is no mechanism for keeping the authenticity and non-repudiation of the transmitting message.

(iv) As security solution on a transport layer, the TLS couldn't provide flexibility for message transmitting, such as encrypted different elements of the message by different key, in which recipients could only read parts of the message about him.

This paper deeply studies the security requirements of Web Service, and then proposes a new security mechanism, a STS-based security architecture, for the Web Services. The rest of this article is organized as follows: section II introduces the Web Services Security Specification briefly. Then we propose a Web Services security Architecture based on STS (Security Token Service) in section III. Finally, we make some conclusions in Section IV.

Manuscript received July 19, 2010.

Kou Hongzhao is with the School of Electronic Technology, Information Engineering University, Zhengzhou, P. R. China 450004; Tel: 86-371-81638701, e-mail: kouhongzhao@ yahoo.com.cn.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

II. WS-Security specifications

The WS-Security specification, which is based on a draft proposed by IBM, Microsoft and Verisign in 2002, is a flexible and feature-rich extension to SOAP to apply security into Web services [1]. The Web Service, which uses the SOAP specification as the message protocol and HTTP as the transport protocol, is to provide mechanisms that will enable business to exchange SOAP messages in a secure environment. The World Wide Web Consortium (W3C) has defined the XML Signature, XML Encryption specifications and Security Assertion Markup Language SAML (XML format Security Token) for digitally signing and encrypting XML-based messages. The WS-Security is a building block used in conjunction with them to accommodate the needs of a set of Web services security specifications.

A. XML Signature

XML Signature is an electronic signature technology which is defined to be used in XML data transmission. XML Signature specification [6], produced by the IETF/W3C, defines electronic signature formats using XML, the creation of electronic signature and rules for the verification. It solves security problems such as authentication, integrity and non-repudiation. The practical benefits of this specification are Partial-Signature and Multiple-Signature. The former allows an electronic signature to be conducted on specific tags in XML data, and the latter enables multiple electronic signatures to be included in one XML document.

The elements in the electronic signature are:

<*SignedInfo*> represents the data that is signed and specifies what algorithms are used. The signature includes the two steps. It firstly calculates the digest of the object, then encrypts the digest with its private key. The object XML signature signed includes not only the reference object, but also other elements such as *CanonicalizationMethod*, *SignatureMethod* element. It is a signature for the entire *SignedInfo* element.

<*CanonicalizationMethod*> indicates the method that is used for canonicalization. The canonical method allows the use of different characters in the XML document. Canonicalization is necessary [2] due to the nature of XML and the way it is parsed by different processors and intermediaries, which can change the data such that the signature is no longer valid but the signed data is still logically equivalent.

<*SignatureMethod*> specifies the algorithm that is used to generate the signature. It is a combination of a digest algorithm and a key dependent algorithm and possibly other algorithms such as padding, for example RSA-SHA1.

<*Reference*> identifies the link of files that is being digested. Multiple <*Reference*> elements may be contained in the <*SignedInfo*> element, and each <*Reference*> element correspond a specific XML data segment at any location. It includes the digest method and resulting digest value calculated over the identified data object. A data object is signed by computing its digest value and a signature over that value. The signature is later checked via reference and signature validation.

<*Transforms*> describes a transformation algorithm used to transform the data before it is digested. It is an optional element and contains a list of one or more Transform elements. Fore example, when the signed object is a binary resource, it is necessary to be converted into base-64 scheme to avoid illegal XML format in the object. There are some other methods such as XPATH and XSLT transformation.

<*DigestMethod*> identifies the algorithm used to calculate the digest, it usually is SHA1. <*DigestValue*> contains the actual base64-encoded digested value of the digest.

<*KeyInfo*> is an optional element that contains information about the key that is needed to validate the signature. It can be omitted if two sides of exchanged message have an agreement on the signature key.

<*Object*> is an optional element that used to define a number of extended information.

B. XML Encryption

XML Encryption is an encryption technology that provides end-to-end security for applications that require secure transmission of XML data. It solves the security problems such as confidentiality, integrity and authentication. The XML encryption provides advanced features as follows:

(i) To encrypt the entire XML document.

(ii) To encrypt an element in the XML file.

(iii) To encrypt particular portion of an element in XML file

(iv) To encrypt the non-XML format resources, such as a JPG format picture.

(v) To encrypt the content that has been encrypted.

For each data object to be encrypted in XML document, the XML Encryption process is:

(i) Select the algorithm (and parameters) to be used in encrypting this data

(ii) Select a key for encryption, if necessary, the information key will be disclosed to the recipient.

(iii) The data must be serialized before it is encrypted.

(iv) Encrypt the serialized data using the algorithm and key from steps (i) and (ii).

(v) Set the Type of the encrypted data (Content or Element).

(vi) Create *EncryptedData* element based on the results and the above options, and replacing the original element with the *EncryptedData*.

For each *EncryptedType* derived element to be decrypted, the decryption process is:

(i) The content of *CipherValue* element will be extracted.

(ii) Obtain the encryption algorithm from the *EncryptionMethod* element.

(iii) Determine the type of encrypted data (Content or Element).

(iv) Obtained the encryption key from the *KeyInfo* key.

(v) Encrypt the ciphertext into original data based on the above information.

III. An STS based Web Services Security Architecture

In order to meet the security requirement for Web Services, the specifications for security are being developed by some organizations and groups. The W3C developed

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

some specifications focused on the XML security [7]; OASIS has published some standards on the access control, such as SAML and XACML [4]; the WS-Security, jointly developed by IBM Corp., Microsoft Corp., VeriSign, is a flexible and feature-rich extension to SOAP to apply security into Web services [1].

Current security mechanisms are not flexible, efficient and manageable enough for Web Service. Each of the above security specifications can only address the security issues on some aspects, such as the WS-Security which only provides mechanisms that will enable business to exchange SOAP messages in a secure environment. However, there is no a widely accepted Web Services security architecture. In the paper, we introduce the Security Token Service (STS) into the WS-Security and present a STS-based security architecture for Web Services, which named as STS-based Secure Web Services (STS-WS).

A. STS-WS Architecture Overview

The STS-WS architecture, shown as Figure 2, is composed of four entities, the UDDI, the Web Service Providers (WSP) with a STS, the Web Service Requester (WSR) and a Certificate Authority (CA).

The CA is used to manage and centrally issue certificates to the entities, and from the CA, a trust domain is established. the STS is a authentication server in service layer, it used to issue, renew, cancel, and validate security tokens for the WSR in a transaction. A group of WSPs within one business shared a STS.

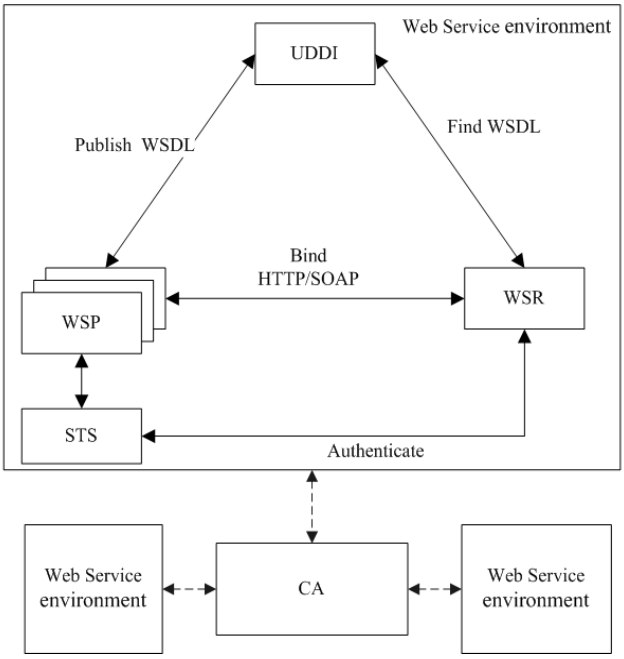

Figure 2 STS-WS Architecture

The WSR is the service user or an entity that has an agreement for the service usage.

B. The Trust Domain

According to X.509 standard, the concept of "trust" is defined as "Generally, an entity can be said to 'trust' a second entity when it (the first entity) makes the assumption that the second entity will behave exactly as the first entity expects" [8]. In a trust domain, all the individuals in the domain complied with the same rules with a common trust anchor (CA). The common trust anchor is typically a root-CA in the domain and all users trust the root-CA. In an organization, WSPs is grouped by business or geography, and a group of WSPs share one STS.

C. STS-based authentication Models

The WSR and the WSP do not attempt to authenticate each other directly. They use STS which validates the WSR's identity and then provides a security token as proof of successful authentication. The authentication model is shown in Figure 3

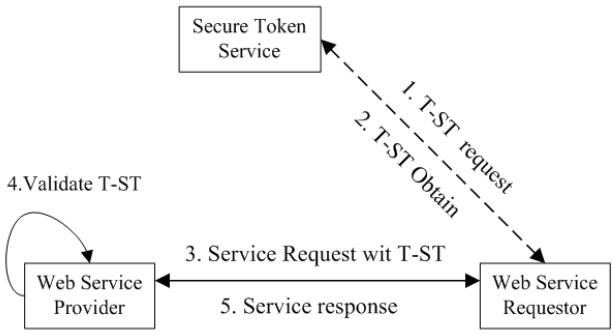

Figure 3 The authentication model

The WSR provides its credentials issued by CA to the STS, and the STS is responsible for the validation of the credentials against an identity store, and then issues a Transaction-specified Security Token (T-ST) to the WSR. The WSR attaches this token to the request and the WSP uses this token to authenticate the WSR.

The CA can be a government agencies responsible for the validation of the person's identity and, in the case of the driver's license, validation of the person's driving skills. The STS validates the WSR's identity, and is only responsible for issuing security token to the WSR according to WSP's secure policy. The ST is transaction-specified and it needs not include the identity of WSR, so the authentication model provides anonymity for the WSR.

D. The mechanism for STS-WS

(i) Registering to the trusted domain

In order to initiate the security mechanism for Web Service, the entities (the UDDI, the WSP, the WSR and STS) in STS-WS must register into the trusted domain firstly, i.e., to get a certificate from the CA in the trust domain. Using the digital certificate issued by CA, entities can conduct operation with the resource in the domain. In the trust domain of Web Services environment, an X.509 digital certificate is packaged as a binary security token transmitting between the entities.

(ii) The Services find to bind

When a WSR wants a Web service, it queries UDDI to find a Web service provider he needed firstly, and then gets the WSDL file of the WSP. The WSR sends UDDI a request message which includes its credential issued by CA. The credential is validated by the UDDI to verify that the it is issued by a trusted CA and that it is not tampered with after it was issued. After the credential is validated, the UDDI sends WSR a Service List as response. While the service requestor selects a service, the UDDI returns the WSDL file of the service with the STS URL, and binds the service to the WSR.

(iii) WSR Obtains Security token

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Before the WSR access the WSP, it must get a transaction ST firstly. When the entities get credential from CA, the STS is trusted by both the WSR and the WSP to provide interoperable security tokens. To obtain the T-ST, the WSR sends an authentication request, with accompanying certificates, to the STS. The STS verifies the certificates presented by the WSR, and then in response, it issues a security token that provides proof that the WSR has authenticated with the STS. The protocol used for issuing security tokens is based on WS-Trust, In order to get *BinarySecuritToken* issued by STS the WSR needs to send a *RequestSecurityToken* message to the STS. The *RequestSecurityToken* contains certificates for the WSR to be authenticated, the required token type and the supported token, then the STS returns a *RequestSecurityTokenResponse* message, which contains security token, such as an XML Security Assertion Markup Language (SAML).

(vi) The security services access

Receiving the WSDL file of the WSP and the transaction T-ST, the WSR can request Web Service according to the service interface and security policy described in the WSDL file. Because the ST associated with a transaction ID, it is valid within the specified transaction.

IV. Conclusions

The existing security specifications for Web Services are developed to meet the security in a particular aspect, such as XML Signature (XMLDSIG) for Message Integrity and Sender/Receiver Identification, WS-Security (WSS) for Securing SOAP Messages and WS-SecurityPolicy for Describing what security features are supported or needed by a Web service. However, there isn't a complete architecture for the Web service security.

In this paper, we introduce the Security Token Service into the WS-Security and present a Security Token Service based security architecture, named STS-based Secure Web Services (STS-WS), for Web Services. The architecture can provide higher security and higher performance services. A prototype system based on the architecture is implemented with the basic functionalities. The next step will be its more perfect, and can interoperate with other systems.

References

[1] OASIS Web Services Security: SOAP Message Security 1.1, OASIS standard specification, 1 February 2006. http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

[2] Sung-Min Lee, O-Sik Kwon, Chan-Joo Oh, et a1. TY * Secure-WS: An intergrated web service security solution based on java. EC-Web ,2003,2738:186-195.

[3] Gerald Brose. A gateway to web services security-securing SOAP with proxies. ICWS-Europe ,2003,2853:101-108.

[4] National Institute of Standards and Technology. Role-based access control-draft 4. http://csrc.nist.gov/rbac/rbac-std-ncits.pdf, 2003.

[5] Ming-Guang Zhang, Wei Qi. E-commerce security system explored. Computer Engineering and Design, 2005,26 (2) :394-396.

[6] Zhang Weiyan, Zhi-Jie Wu, Xia Tao. Web Services messages in Communication Research. Computer Engineering and Design, 2005,26 (10) :2621-2623.

[7] XML Encryption Syntax and Processing. Technical report, W3C, December 2002. http://www.w3.org/TR/xmlenc-core/.

[8] Itu-t recommendation x.509. information technology. open systems interconnection-the directory: Public-key and attribute certificate frameworks.

[9] National Institute of Standards and Technology, Guide to Secure Web Services, http://csrc.nist.gov/publications/nistpubs/800-95/SP800-95.pdf

[10] L. Lo Iacono and J. Wang, "Web service layer security (WSLS)," Network Security, vol. 2, pp. 10–13, 2008.

ISBN: 978-988-17012-0-6
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)