A Stream based Implementation of Secured SOA Model using XML Encryption and XML Signature

Srinath K S[1] Mallamma C G[2] Shankar Rana[3] Vijay Kumar F G[4]

[1,2,3,4] Sambhram Institute of Technology
Bangalore-560097
Karanataka, India

ABSTRACT
Web service security is essential for SOA-based applications; it has explorative set of technologies such as Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL) and Universal Description, Discovery and Integration (UDDI), and Electronic Business XML (ebXML). These promote an environment for organizations to communicate in Internet. The inevitable challenge that organizations face today is to implement adequate Web Service Security as the Web Service transactions are done mainly through plain text formats, making them easy to get hacked. This paper proposes the XML signature and encryption as the core of Service Oriented Architecture (SOA) for web service security, and describes how to create and verify XML signature, and how to encrypt and decrypt XML data. This application provides security based on the parameters such as confidentiality, integrity, authentication and authorization.

Keywords
Web Services, Service Oriented Architecture (SOA), Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), Encryption, XML Signature.

1. INTRODUCTION
SOA brought a direct consequence of specific business and technology drivers that came into existence over the past decade. The outsourcing of noncore operations and the importance of business process reengineering have been key influences driving the surfacing of SOA as an important architectural approach to business information technology today.

Web Services and SOAs are often considered to be the most important technological innovations [1]. A web service is any service that is available over the Internet, uses a standardized XML messaging system, and is not tied to any operating system or programming language [2].

The paper describes the techniques developed to secure SOA architecture for efficient, effective and secure communication. The techniques described include XML Encryption and XML Signature, which are implemented over SOAs using Web Services technologies such as SOAP and WSDL.

1.1 Web Service Technology Stack

Web service stack shown in the Fig: 1 categorizes the technology of web service into a layered model. The stack starts at the service transport with the basic technologies that allow data transfer from one machine to another. Each layer is built on the lower layers and adds higher-level abstractions.

Layer description	Implementation	Other concerns			
Standard messaging	ebXML	Quality of service	Management	Security	Service development
Service composition	BPEL4 WS				
Service Registry	UDDI, ebXML				
Service description	WSDL				
Service messaging	SOAP/XML				
Service transport	HTTP, FTP, SMTP				

Fig 1: Web services technology stack

The upper layers of the stack do not necessarily depend on the lower layers and in some ways are orthogonal concerns. They are shown in this format to demonstrate the higher level of abstraction. Thus, the security may be guaranteed by the safety mechanism of present network level.

Although SOAP and HTTP are enough for interoperability XML messages transmission and WSDL describes the service, but complete demand of security coverage in electronic commerce and so on must need more security considerations.

2. WEB SERVICE TECHNOLOGIES
Several technologies have been introduced under the web service rubric and many more will be introduced in coming years. The web service paradigm has grown so quickly, hence several competing technologies are attempting to provide the more capability techniques [4]. However, the web service vision of seamless worldwide business integration will not be feasible unless the core technologies are supported by every major software company in the world.

Over the past two years, three primary technologies have emerged as worldwide standards that make up the core of today's web services technology such as SOAP, WSDL and UDDI.

2.1 Simple Object Access Protocol (SOAP)

For SOA architectures, the de facto standard for the basic protocol for communication between two parties is the SOAP. Because SOAP is the basic messaging protocol for Web services, other specifications are built on top of the SOAP specification.

2.2 Web Service Description Language (WSDL)

WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. WSDL describes the interface of a web service in a standardized way. WSDL standardizes how a web service represents the input and output parameters of an invocation externally, the structure of function, the nature of the invocation and the service protocol binding. WSDL allows unperceptive clients to automatically understand how to interact with a web service.

2.3 Universal Description Discovery Integration (UDDI)

The UDDI specification defines a platform-independent framework for businesses to describe their publicly available services, discover other services, and share information about points of interaction in a global registry. UDDI is a building block that enables organizations to quickly find and conduct business transactions with other organizations using open standards.

The UDDI specification is an attempt for organizations to accomplish the following:

Discover the right partner to conduct business with one out of the millions that exist on the Internet today

Create an industry-accepted standardized approach to reach partners and customers with information on their services and convey the preferred method for integration between disparate systems

Characterize how business transactions are used in commerce, once a preferred partner is selected through electronic means.

UDDI can be the solution to many business problems. By simplifying B2B interactions, a business can discover other business, independent of the choice of standards and protocols. The data captured within UDDI is divided into three main categories:

White pages

This includes general information about a specific company. For example, business name, business description, contact information, address and phone numbers.

Yellow pages

This includes general classification data for either the company or the service offered. For example, this data may include industry, product, or geographic codes based on standard taxonomies.

Green pages

This category contains technical information about a web service. Generally, this includes a pointer to an external specification and an address for invoking the web service. UDDI is not restricted to describing web services based on SOAP. Rather, UDDI can be used to describe any service from a single web page or email address all the way up to SOAP, CORBA, and Java RMI services.

3. SERVICE-ORIENTED ARCHITECTURE

SOA is an architectural style based on loosely coupled and interoperability software components that provide services. A service is a functionality availed by a service provider in order to deliver the response for consumer's request. The service provider returns a response to the service consumer containing the expected response.

A service-oriented architecture is the underlying structure supporting communications between entities. SOA defines how two computing entities, such as programs, interact in a way so as to enable one entity to perform a unit of work on behalf of other entity.

SOA configures entities (services, registries and contracts) to maximize loose coupling and reuse. In Fig 2 it allows the consumer of a service to ask a third-party service broker for available services. If the registry has such a service, it gives the consumer a contract and the binding address for the service. SOA consists of the following six entities configured together to support the find, bind, and interact.

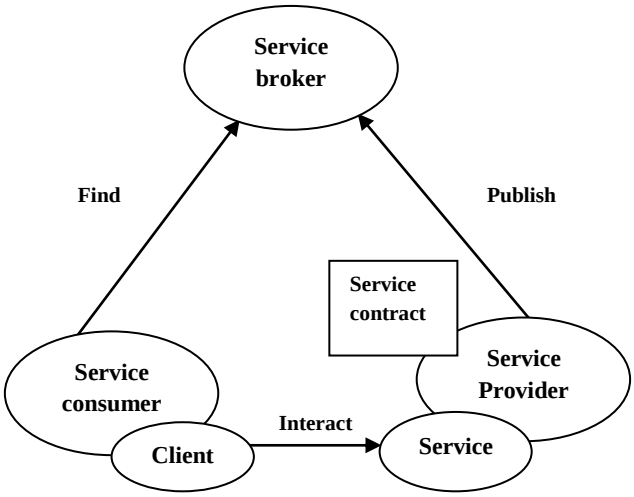

Fig 2: Service Oriented Architecture

3.1 Service Consumer

The service consumer is a software module that requires a service. It is the entity that initiates the locating of the service in the registry, binding to the service with the help of WSDL file which is obtained from registry, and interacting with service function by sending SOAP request. The service consumer executes the service by sending a SOAP request formatted according to the contract.

3.2 Service Provider

The service provider is the software module that accepts and executes SOAP requests from consumers. It can be a mainframe system, a component, or some other type of software system that executes the service request. The service provider publishes its contract in the registry for access by service consumers.

3.3 Service Registry

A service registry is a resource that provides a controlled access to the web service. In effect, it is a constantly evolving catalog of information about the available services in an SOA implementation. This entity accepts and stores the contracts from service providers and provides those contracts to interested service consumers.

3.4 Service Contract

Service contract is a specification for the consumer of a service for interacting with the provider of the service. It specifies the format of the request and response from the services. A service contract contains a set of pre conditions and post conditions. The pre conditions and post conditions specify the state that the service must be in to execute a particular function. The contract may also specify quality of service (QoS) levels. QoS levels are specifications for the nonfunctional aspects of the service. For instance, a quality of service attribute is the amount of time it takes to execute a service method.

4. XML ENCRYPTION

XML Encryption is a specification recommended by the W3C for securing its content. It is a flexible way to create common information formats and share both the format and the data on the Internet, Intranets, and private networks. It is a formal recommendation from the World Wide Web Consortium, similar to the language of today's web pages, the Hyper Text Markup Language (HTML) [8], [9].

The purpose of XML Encryption is to maintain the confidentiality of information during the encryption process with the help of Secure Socket Layer (SSL) or Transport Layer Security (TLS). The encryption process is controlled by an encryption key as shown in the Fig 3. In a more general approach, XML Encryption provides end-to-end security and is used to encrypt the XML document which further represents the encrypted data in XML documents. This is possible only when proper use of algorithms and technologies are defined. Standard algorithms like DES, AES, RSA and many more support the strong encryption. [10].

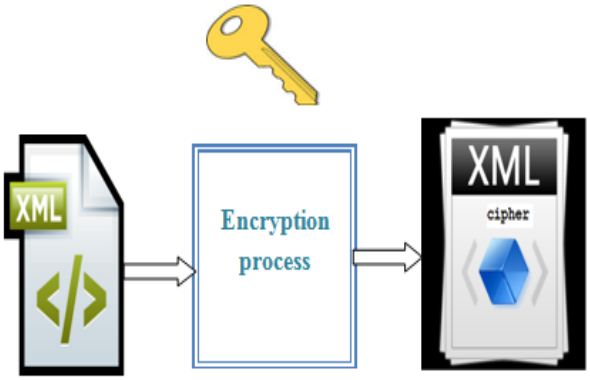

Fig 3: XML Encryption

5. XML DECRYPTION

XML Decryption is used for the conversion of cipher text i.e. encrypted XML file to the original message i.e. plain text which is shown in the Fig 4. During the process, the conversion of messages takes place by using decryption key. Same key should be used to decrypt the cipher text to obtain back the original message or file [11], [12].

Fig 4: XML Decryption

6. XML SIGNATURE

XML Signatures are used for signing the digital content and verifying the digital signatures. This method is used to provide data integrity, so that no one can tamper with the information. XML Signatures are designed for secure transactions in XML format.

The XML Signature supports any type of digital signature encryptions using all possible standard encryption algorithms. MD5, SHA-1 and RSA are some of the algorithms that are used to calculate the hash values of the data. The signature process is carried out on the hash values. After the hash value is signed, it is guaranteed that the integrity of the original document cannot be changed; Fig 5 shows the signed XML document.

7. DESIGN AND IMPLEMENTATION

This work is based on the encryption and signature techniques to provide security to the XML data being transferred over the HTTP in the form of XML files.

As transferring the XML files without any security makes it vulnerable to attacks such as Data tampering, Eavesdropping and Man-in-the-middle attack, therefore, security measures are to be provided for data integrity, confidentiality and client/server authentication [6], [7]. Fig 6 shows the architecture of proposed system which includes service provider, service consumer, registry, and signer/validation modules.

7.1 Encryption and Encoding

The security is provided to the XML document by applying encryption technique. This provides the confidentiality to data being transferred over network. To achieve this, the SAXParserFactory and DocumentHandler are used to parse the entire input file which is in the XML format.

The parser reads input file in the order of elements such as, for example, <start> is the start element of any file and its corresponding end element is </start>. The entire data present in between these elements are considered as characters and are parsed accordingly. Each element is encrypted using DES Algorithm by providing a key, which is then encoded in the Base64 format because special characters can't be transferred over the network.

Each encrypted element is bound by <cipher> as their start element and </cipher> as their end element. Similarly, the entire XML file is enveloped in <st> and </st> as their start and end document respectively. This file is, then, transferred to the receiver end that decodes and decrypts it using the same key and algorithm, thus obtaining the file similar to the input file.

```
<student>
<name>niru</name>
<address>bangalore</address>
<usn>123</usn>
<Signature
xmlns="http://www.w3.org/2000/09/xmldsig
#"><SignedInfo>
<CanonicalizationMethod
Algorithm="http://www.w3.org/TR/2001/REC
-xml-c14n-
20010315"></CanonicalizationMethod>
<SignatureMethod
Algorithm="http://www.w3.org/2000/09/xml
dsig#rsa-sha1"></SignatureMethod>
<Reference URI=""> <Transforms>
<Transform
Algorithm="http://www.w3.org/2000/09/xml
dsig#enveloped-signature"></Transform>
</Transforms><DigestMethod
Algorithm="http://www.w3.org/2000/09/xml
dsig#sha1"></DigestMethod><DigestValue>f
ekhE6bWEDrbzSWUvNrNd3e8Fno=</DigestValue
> </Reference> </SignedInfo>
<SignatureValue>
BmzmcG8pnX3i1Fy7unaC811hvnO7V/UgFdNf2Fzk
ZWoC/LHoTXkY7bnVfzlnTacs4ny/dm6PvO1a
lLrr1BX9rHujLfUOqB7c4YmY34NlqFqDeOhofXot
NiWi/Ok51Hg2Gs0eMUWIlk5dfg9uAJ/yaZRq
ss2wyl4Xz+SPi4dl5/c=</SignatureValue>
<KeyInfo> <X509Data> X509Certificate>
MIICNTCCAZ4CBFFne3UwDQYJKoZIhvcNAQEEBQAw
YjEQMA4GA1UEBhMHVW5rbm93bjEQMA4GA1UE
CBMHVW5rbm93bjEQMA4GA1UEBxMHVW5rbm93bjEK
MAgGA1UEChMBYjEKMAgGA1UECxMBYTESMBAG
A1UEAxMJMTI3LjAuMC4xMB4XDTEzMDQxMjAzMTE0
OVoXDTEzMDcxMTAzMTE0OVowYjEQMA4GA1UE
BhMHVW5rbm93bjEQMA4GA1UECBMHVW5rbm93bjEQ
MA4GA1UEBxMHVW5rbm93bjEKMAgGA1UEChMB
YjEKMAgGA1UECxMBYTESMBAGA1UEAxMJMTI3LjAu
MC4xMIGeMA0GCSqGSIb3DQEBAQUAA4GMADCB
iAKBgE1Rpygs/ncvcjJkZkC0W9tuK6NK74JJUEtH
gc3uvgtdmXzYnyGWnLds+hMelRajS2t9kC8d
ALfQLWoJjwb6vlQbxV21UN8lVZvDX67Cm7LJJCem
o7iTwmWi1h3GfVDQjbqvHwyk9xdQhtMFz5DP
Fd+tllyLbetawE1Pi5a1avdvAgMBAAEwDQYJKoZI
hvcNAQEEBQADgYEAAkfkbhlyVNN/TCnUvxJa
T3EorvPlgWKtJkIejCGysTm2RyLfg18VYsmWyfgJ
VVqzkNiMxg/6d0j5ngVFJkemXqH0/bD1Ghzw
3XE+om80xmk4U0Wr4VWHds3c6i2ud9+AL1inCpot
uPbfyH9gF48BYlHg/mpJ68vRI0F+llshgoI=</X5
09Certificate>
/X509Data><KeyValue><RSAKeyValue><Modulu
s>TVGnKCz+dy9yMmRmQLRb224ro0rvgklQS0eBze
6+C12ZfNifIZact2z6Ex6VFqNLa32QLx0At9At
agmPBvq+VBvFXbVQ3yVVm8NfrsKbsskkJ6ajuJPC
ZaLWHcZ9UNCNuq8fDKT3F1CG0wXPkM8V362W
XItt61rATU+LlrVq928=</Modulus>
<Exponent>AQAB</Exponent></RSAKeyValue>
</KeyValue></KeyInfo></Signature></stude
nt>
```

7.2 SOA Interaction

SOA Architecture consists of provider, registry and consumer. The registry acts as the server and both the consumer and provider act as clients. The provider initially encrypts, encodes and places its WSDL file on the registry database. The registry stores it in the similar form and waits for its request from the consumer.

The consumer sends the file name required, and the server searches for it in its database. If available it sends that file to the consumer else it returns appropriate message. The consumer, on receiving the file, decodes and decrypts the file using the same key and algorithm as that of the ones used in provider.

7.3 SOAP Messages

The provider and consumer interacts directly using SOAP request and response message. Both the provider and consumer act as client and server.

In this design, the consumer initially encrypts and encodes a SOAP request and as being the client sends it to the provider which acts as the server. The provider, on receiving the request, decodes, decrypts the request and sends an encrypted and encoded SOAP response back to the client.

7.4 Signature

The messages sent between the sender and the receiver is signed to provide authenticity and integrity. A third party acts as the signer and validator for these messages.

Initially the sender acts as a client and sends its data to be signed to the signer along with another file which contains a code verifying its authenticity. The signer compares the code with the code in its database, if found it uses DSA-SHA1 algorithm to generate a signed file where the signature is enveloped and passes this signed file back to the sender. If the code is not found, it does not sign the data and returns a message for the same to the sender.

Then, the sender sends the signed file to the receiver which now acts as the server. The receiver stores the signed file in its database. It then passes this signed file to the validator for validating the received file. The receiver is the client and validator is the server, which validates and sends a text “valid signature” in case there is no tampering with the data, else “invalid signature” if any part of the data is modified in transaction.

8. RESULTS

The results generated based on the proposed model are tabulated. Table 1 shows the results obtained based on the number of tags used in an input file (XML document) and the time taken for encryption and decryption (in milliseconds).

The graph from Fig 7 implies that if the size of the xml document (number of tags) increases, then the time required for the encryption and decryption also increases. The time taken by encryption and decryption module to encrypt and decrypt the XML document is found almost the same or nearer.

The time taken for signer and validation module for signing and validating the XML document are tabulated in table 2. Figure 8 shows and concludes that the time taken for signing is more than the time taken for validation.

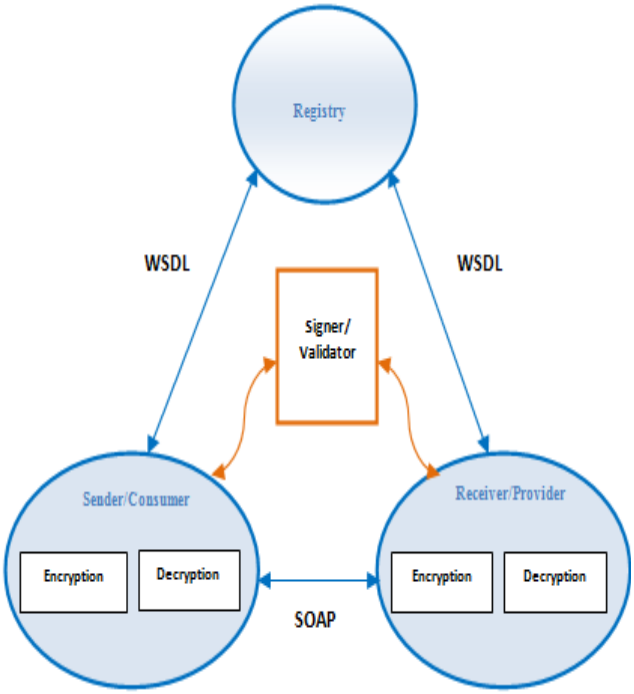

Fig 6: System Architecture

Table 1: Encryption and Decryption Time

No. of tags	Encryption time(ms)	Decryption time(ms)
50	890	880
100	1115	1027
150	1227	1220
200	1506	1499
250	1517	1512
300	1765	1702

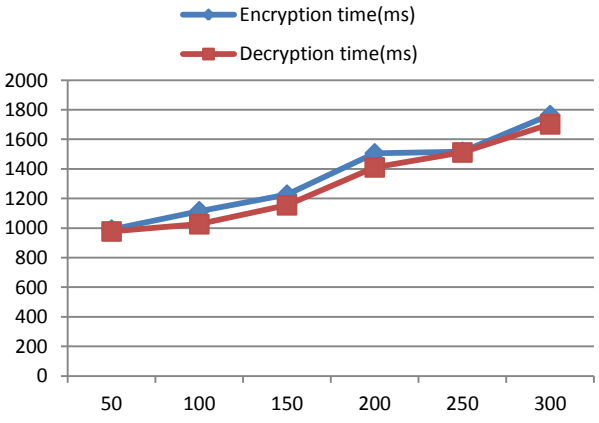

Fig 7: Time profile for Encryption and Decryption

Table 2: Signing and Validation Time

No. of tags	Signature time (ms)	Validation time (ms)
50	487	300
100	560	370
150	580	400
200	605	420

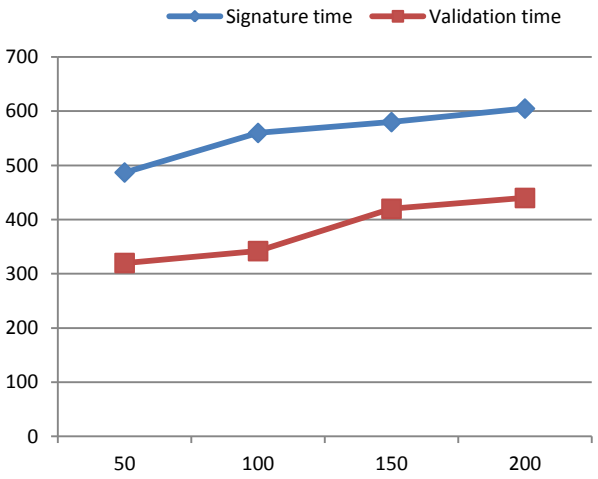

Fig 8: Time profile for Signing and Validation

9. CONCLUSION

In this paper, a stream based XML Encryption and XML Signature is implemented to overcome from different type of attacks. The outcome result assures security for SOA architecture from various attacks. Further, design can be upgraded to provide security against attacks like Denial of Service (DoS) and Replay attacks by implementing the standard Access control and WS-Policy.

10. REFERENCES

[1] Jensen, M., Gruschka, N., Herkenh¨oner, R., and Luttenberger, N., (2007), "SOA and Web Services: New Technologies, New Standards – New Attacks" Fifth European Conference on Web Services, 0-7695-3044-3/07, 2007.

[2] Gu Yue-sheng, Ye Meng-tao, Gan Yong, "Web Services Security Based on XML Signature and XML Encryption" Journal of Networks, Vol. 5, No. 9, 2010.

[3] James McGovern, Sameer Tyagi, Michael E. Stevens and Sunil Mathew, Java Web Services Architecture, Morgan Kaufmann, 2003.

[4] Esmiralda Moradianvand Anne Håkansson, Possible attacks on XML Web Services, IJCSNS International Journal of Computer Science and 154 Network Security, VOL.6 No.1B, January 2006, pp 154-170.

[5] Ethan Cerami, Web Services Essentials, O'Reilly, First Edition, February 2002.

[6] Alonso, G. Casati, F. Kuno, H. Machiraju, V. "Web Services: Concepts, Architectures and Applications", Springer, 2004.

[7] T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, 2005.

[8] Alvarez, G. Petrovic, S. "A new taxonomy of Web attacks suitable for efficient encoding. Computer & Security", Vol.22 (5), 2003, p435-449.

[9] RA. K. Saravanaguru, George Abraham, Krishnakumar Venkatasubramanian, Kiransinh Borasia (2011) "Securing Web Services Using XML Signature and XML Encryption.

[10] Takeshi Imamura, Andy Clark, Hiroshi Maruyama "A Stream-based Implementation of XML Encryption", pp 11-17.

[11] J.Hanson (2005) "Managing XML Encryption with Java", Devx Website - http://www.devx.com/xml/Article/28701/1763.

[12] T. Imamura, A. Clark, and H. Maruyama, "A Stream-Based Implementation of XML Encryption," Proc. ACM Workshop XML Security (XMLSEC '02), pp. 11-17, 2002. [9]

[13] "XML Digital Signatures (Cover Pages hosted by OASIS) (Technology Reports)" - http://xml.coverpages.org/xmlSig.html.

[14] "XML Security: Signature, Encryption and Key Management (W3C note)" - http://www.w3.org/2004/Talks/0520-hhxmlsec/