ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 2, Issue 4, April 2013

A Survey on Web Services Security and Its Specifications

Mrs. K. Rajasri, Mrs. S. Tamilarasi , S. Sathiyadevi

***Abstract*— Web services are being effectively used for interoperable solutions transversely a variety of industries. One of the type reasons for attention and outlay in Web services is that Web Services are well-suited to enable service-oriented systems. An XML-based technology that is SOAP, XML Schema and WSDL offer a mostly-accepted establishment on which to construct interoperable Web services. Security is important for any distributed computing environment. But, security is becoming even more important for Web services due to some reasons. In this survey of WS-Security and its specifications is discussed. Some of the specifications are discussed with architecture and the issues of the Ws-Security also described. Performance of the WS-Security is analysed by using message types.**

.

***Index Terms*—Security, Web Services, WS-Security**

I. INTRODUCTION

Web services widen the World Wide Web environment to offer the way for software to connect to further software applications. Applications access Web services via ubiquitous Web protocols and data formats such as HTTP, XML, and SOAP, with no need to worry about how each Web service is implemented. Web services combine the best aspects of component-based development and the Web, and are a cornerstone of the Microsoft .NET programming model. Web Services can convert your application into a Web-application, which can publish its function or message to the rest of the world. A web service is a software function provided at a network address over the web or the cloud, it is a service that is "always on" as in the concept of utility computing. Providing security to web services is achieved by WS-Security. WS-Security describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication. These mechanisms can be used to accommodate a wide variety of security models and encryption technologies.

WS-Security also provides a general-purpose mechanism for associating security tokens with messages. No specific type of security token is required by WS-Security. It is designed to be extensible (e.g. support multiple security token formats). For example, a client might provide proof of identity and proof that they have a particular business certification. Additionally, WS-Security describes how to encode binary security tokens. Specifically, the specification describes how to encode X.509 certificates and Kerberos tickets as well as how to include opaque encrypted keys. It also includes extensibility mechanisms that can be used to further describe the characteristics of the credentials that are included with a message. This specification is intended to provide a flexible set of mechanisms that can be used to construct a range of security protocols; in other words this specification intentionally does not describe explicit fixed security protocols. As with every security protocol, significant efforts must be applied to ensure that security protocols constructed using WS-Security are not vulnerable to a wide range of attacks. If you choose to deploy Web services, security will be a major issue. When deploying a Web service, you have to think about how you will secure that service.

Security encompasses the following:

1. Authentication: promises that the service is accessible for anybody with a verified identity.
2. Authorization: promises that the authenticated individuals have the right to access the service or data.
3. Confidentiality: promises that the data approved between the requester and provider is confined from eavesdroppers.
4. Integrity: Provides that the message was not modified in its pathway from requestor to provider.
5. Non-repudiation: promises that the sender of the message could not deny that he/she sent it at a afterward position in time.
6. Accessibility: make sure that the service is for all time accessible and that it is not damaged by attacks, similar to denial-of-service (DoS), from exterior or in the interior of the system hosting the service.

WS-Transaction is one of a series of specifications from an industry group that includes IBM, Microsoft, and BEA Systems. The WS-Transaction interface defines what constitutes a transaction and what will determine when it has completed successfully. Each transaction is part of an overall set of activities that constitute a business process that is performed by cooperating Web services. The overall business process is formally described using the Business Process Execution Language (BPEL). WS-Coordination is a companion specification that defines the context and exactly how information is exchanged during the business process.

The WS-Transaction specification is an activity of the Web Service Interoperability Organization (WS-I

***Mrs. K. Rajasri**, Assistant Professor, Department of Computer Science, Christ College of Engineering and Technology, Pondicherry University Pondicherry, India,*
***Mrs. S. Tamilarasi**, Department of Computer Science, Christ College of Engineering and Technology, Pondicherry University, Pondicherry, India,*
***S. Sathiyadevi**, Department of Computer Science, Christ College of Engineering and Technology, Pondicherry University, Pondicherry, India,*

Organization) which is an industry-wide effort at standardizing how Web services are requested and delivered.

E-business relies on the exchange of information between business partners over a network. In such a setup, as the information/data travels from the source to the destination, there is always the risk of the data being stolen or modified. The same security risks are applicable to web services transactions. In web services transactions using SOAP, the data are passed between the service invoker and service provider as plain XML, so anyone who intercepts the messages can read the data that are exchanged. In the web services scenario the security aspect is more complicated because:

- Soap messages can be sent using different transport applications or protocols like HTTP, SMTP, etc., which might have a very high security model to no security model at all. Hence, there is a need for a comprehensive security framework for web services applications, which is common to all types of transport protocols.
- There could be legitimate intermediaries that might need to access a part or whole of the SOAP message, and even modify the message. Thus the security model must be comprehensive enough to allow such intermediaries.

II. SURVEY ON WS-SECURITY

WS-Security or **Web Services Security is** a flexible and characteristic loaded addition to SOAP to be relevant security to web services. It is a associate of the WS-* family of web service specifications and was published by OASIS. (OASIS stands for Organization for Advancement of Structured Information Standard)

The protocol identifies how integrity and confidentiality can be compulsory on messages and lets the communication of diverse security token formats, for instance, SAML, Kerberos, and X.509. Its major focal point is to make use of XML Signature and XML Encryption to offer end-to-end security.

WS-Security describes three main mechanisms:

- SOAP messages to reassure integrity. Signed messages too provide non-repudiation.
- Encrypt SOAP messages to reassure confidentiality.
- Attach security tokens to determine the sender's identity.

The specification permits various signature formats, encryption algorithms and several trust domains, and is unlock a variety of securities token models, for instance:

- X.509 certificates,
- Kerberos tickets,
- UserID/Password credentials,
- SAML Assertions, and
- Custom-defined tokens.

The token formats and semantics are described in the related profile documents.

WS-Security integrates security characteristics in the header of a SOAP message, functioning in the application layer. These methods by themselves do not offer an entire security solution for Web services. As a substitute, this requirement is a structure chunk that can be used in conjunction with further Web service extensions and higher-level application-specific protocols to contain a broad various security models and security technologies. In common, WSS by itself does not offer any assurance of security. While implementing and utilizing the framework and syntax, it is about to the implement or to make sure that the result is not susceptible.

Key management, trust bootstrapping, federation and agreement on the technical details (ciphers, formats, and algorithms) are exterior the extent of WS-Security.

Back-to-Back security

If a SOAP mediator is required, and the mediator is not or is not as much of trusted, messages require to be signed and optionally encrypted. This may be the case of an application-level proxy at a network perimeter that will end TCP connections.

Non-Repudiation

The standard method for non-repudiation is to write transactions to an audit trail that is subject to specific security safeguards. However, if the audit trail is not sufficient, digital signatures may provide a better method to enforce non-repudiation. WS-Security can provide this.

Substitute Transport Bindings

Though approximately all SOAP services employ HTTP bindings, in presumption further bindings such as JMS or SMTP could be utilized; in this case back-to-back security would be required.

Reverse proxy/common security token

Still if the web service relies ahead transport layer security, it may be required for the service to recognize about the end user, if the service is relayed by a (HTTP-) reverse proxy. A WSS header can be used to communicate the end user's token, promised for by the reverse proxy.

A. Issues

- If there is a repeated message exchanges flanked by service provider and consumer, the overhead of XML SIG and XML ENC are important. If back-to-back security is necessary, a protocol similar to WS-SecureConversation may decrease the overhead. If it's enough, employ simply encryption *or* signing, as the grouping of both is considerably slower than the simple sum of the single operations.
- The inclusion of several XML schemata like SOAP, SAML, XML ENC, and XML SIG may cause dependencies on dissimilar versions of library functions like canonicalization and parsing, which are hard to handle in an application server.
- If merely CBC mode encryption/decryption is applied or if the CBC mode decryption is applied without confirm a secure checksum (signature or MAC) previous to

decryption next to the implementation is probable to be weak to padding oracle attacks.

B. Performance

WS-Security inserts important overhead to SOAP processing appropriate to the increased size of the message on the wire, XML and cryptographic processing, need quicker CPUs and additional memory and bandwidth.

An evaluation in 2005[2] measured 25 types of SOAP messages of different size and complexity processed by WSS4J with both WS-Security and WS-SecureConversation on a Pentium 4/2.8 GHz CPU. Some findings were:

- Encryption was more rapidly than signing.
- Encryption and signing jointly were 2–7 times slower than signing only and produced considerably bigger documents.
- Depending on the type of message, WS-SecureConversation also completed with no difference or reduced processing time by semi in the best case.
- It get fewer than 10 milliseconds to sign or encrypt up to an array of 100 kilobytes, but it obtain about 100~200 to perform the security operations for SOAP.

III. SPECIFICATIONS IN WS-SECURITY

The following describes the specifications in detail.

- **WS-Policy** symbolizes a set of specifications that explain the capabilities and constraints of the security policies on mediators and endpoints (e.g. necessary security tokens, supported encryption algorithms, privacy rules) and how to relate policies with services and endpoints.
- **WS-Trust** explains a framework for trust models that allows Web services to securely interoperate by requesting, issuing, and exchanging security tokens.
- **WS-Privacy** will portray a replica for operation of Web services and requestors' condition confidentiality preferences and organizational confidentiality observe statements.
- **WS-SecureConversation** explains how to control and authenticate message exchanges between end users, counting security context exchanges and creating and deriving session keys.
- **WS-Federation** illustrates how to handle and broker the trust relationships in a varied federated environment, counting support for federated identities, distribution of attributes, and management of pseudonyms.
- **WS-Authorization** will depict how to deal with authorization data and authorization policies.

A. WS-Policy

The WS-Policy and WS-PolicyAttachment specifications broaden this establishment and recommend mechanisms to characterize the capabilities and necessities of Web services as Policies.

Service metadata is a term of the observable features of a Web service, and consists of a combination of machine- and human-readable languages. Machine-readable languages facilitate tooling. For instance, tools that use service metadata can mechanically produce client code to call the service. Service metadata can explain dissimilar elements of a Web service and therefore facilitate special levels of tooling hold up.

First, service metadata can explain the plan of the payloads that a Web service sends and receives. Tools can utilize this metadata to mechanically produce and authenticate data sent to and from a Web service. The XML Schema language is commonly used to explain the message exchange format in the SOAP message construct, i.e. to symbolize SOAP Body children and SOAP Header blocks.

Second, service metadata can explain the 'how' and 'where' a Web service exchanges messages, i.e. how to characterize the real message format, what headers are utilized, the transmission protocol, the message exchange pattern and the listing of obtainable endpoints. The Web Services Description Language is presently the majority common language for recounting the 'how' and 'where' a Web service exchanges messages. WSDL has extensibility points that can be utilized to develop on the metadata for a Web service.

Third, service metadata can explain the capabilities and necessities of a Web service, i.e. representing whether and how a message have to be secured, whether and how a message have to be delivered reliably, whether a message have to flow a transaction, etc. Revealing this class of metadata about the capabilities and requirements of a Web service allows tools to produce code modules for engaging these behaviors. Tools can employ this metadata to make sure the compatibility of requestors and providers. Web Services Policy can be utilized to symbolize the capabilities and necessities of a Web service.

Web Services Policy is a machine-readable language for representing the capabilities and requirements of a Web service. These are called 'policies'. Web Services Policy presents mechanisms to symbolize reliable grouping of capabilities and requirements, to establish the compatibility of policies, to name and reference policies and to associate policies with Web service metadata constructs such as service, endpoint and operation. Web Services Policy is a easy language that has four elements - Policy, All, ExactlyOne and PolicyReference - and one attribute - wsp: Optional.

B. WS-Trust

WS-Trust is supports on a method in which a Web service can need that an incoming message proves a rest of claims (e.g., name, key, permission, capability, etc.). If a message enters without having the required evidence of claims, the service is supposed to take no notice of or refuse the message. A service can specify its required claims and connected information in its policy as explained by [WS-Policy] and [WS-PolicyAttachment] specifications.

Authentication of requests is depends on a blend of elective network and transport-offered security and information confirmed in the message. Requestors can authenticate receivers using network and transport-offered security, claims verified in messages, and encryption of the request using a key well-known to the receiver.

One method to reveal authorized utilize of a security token is to contain a digital signature using the linked secret key (from a proof-of-possession token). This permits a requester to establish a necessary set of claims by associating security tokens (e.g., PKIX, X.509 certificates) with the messages. If the requester does not have the essential token to establish necessary claims to a service, it can make contact with suitable authorities and demand the wanted tokens with the appropriate claims. These "authorities", which we submit to as security token services, may in revolve, need their individual set of claims for authenticating and authorizing the request for security tokens. Security token services form the foundation of trust by issuing a variety of security tokens that can be used to broker trust relationships between dissimilar trust domains.

This specification also defines a common method for multi-message exchanges during token acquisition. One instance utilize of this is a brave response protocol that is also defined in this specification. This is utilized by a Web service for added confront to a requester to make sure message newness and verification of authorized use of a security token.

WS-Trust Architecture

Figure 1 shows a common outline of the WS-Trust architecture.

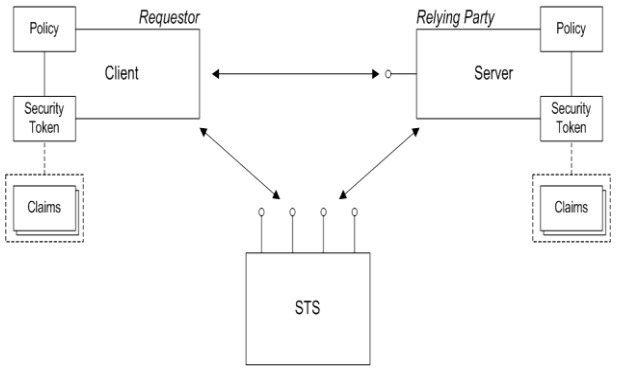

Fig 1: WS-Trust Architecture

Requestor

A requester is an entity that attempts to invoke a secure operation above a network connection. Put into practice, a requester is normally a Web service client.

Relying Person

A relying Person refers to an entity that have various services or resources that have to be secured next to unauthorized access. In carry out, a relying Person is classically a Web service.

Security Token

A security token is a set of security data that a requester sends within a request so as to invoke a secure operation or to get access to a secure resource. In the WS-Trust framework, the idea of a security token is somewhat common and can be used to explain any chunk of security data that might escort a request.

In opinion, WS-Trust can be utilized with the following kind of security token:

- SAML token.
- UsernameToken token.
- X.509 certificate token.
- Kerberos token.

SAML token

A SAML token is a mainly flexible type of security token. The SAML specification describes a common reason XML schema that allows you to enfold approximately any type of security data and enables you to sign and encrypt element or the entire token. SAML is a well-liked option of token to employ in the framework of WS-Trust, because SAML has all of the essential features to hold up characteristic WS-Trust authentication scenarios.

Claim

A SAML security token is officially defined to consist of a set of claims. Every claim characteristically holds an exacting type of security data.

Policy

In WS-Trust scenarios, a policy can symbolize the security configuration of a participant in a secure application. The requester, the relying person, and the security token service are all configured by policies. For instance, a policy can be used to arrange what types of authentication are sustained and necessary.

Security Token Service

The security token service (STS) put down at the heart of the WS-Trust security architecture. In the WS-Trust typical, the subsequent bindings are definite:

- Issue binding—the specification describes this binding as follows: Based on the credential offered/established in the request, a novel token is issued, perhaps with new proof information.
- Validate binding—the specification describes this binding as follows: The scope of the specified security token is estimated and an effect is returned. The effect may be a status, a novel token, or both.
- Renew binding—the specification describes this binding as follows: formerly issued token with running out is obtainable and the similar token is returned with novel running out semantics.
- Cancel binding—the specification describes this binding as follows: When a formerly issued token is no longer wanted, the Cancel binding can be utilized to terminate the token and terminating its use.

C. WS-Privacy

Organizations can make use of WS-Privacy to exchange their privacy policies and ensure to observe whether requesters mean to fulfill with these policies. WS-Privacy is a combination of WS-Policy and WS-Trust.

D. WS-SecureConversation

WS-Security offers methods for securing a particular message in a one-way message exchange. Frequently interactions between a Web service and a consumer will result in multiple messages being exchanged. While each message could be secured in separation, it is more proficient to set up various form of context that the Web service and customer share and employ that context to lessen the trouble with respect to securing each message exchanged. WS-SecureConversation describes a Security Context Token to execute this task. WS-SecureConversation, also called Web Services Secure Conversation Language, is a specification that offers secure communication stuck between Web services using session keys. WS-SecureConversation, released in 2005, is an extension of WS-Security and WS-Trust.

WS-SecureConversation workings by defining and executing an encryption key to be shared amongst all the entities concerned in a communications session. The originating entity characterizes the encryption algorithm and produces the key, which is embedded in a SOAP (Simple Object Access Protocol) message. (This key preserve also is used to encrypt the SOAP message itself.) When the proposed destination entities obtain the message, they decrypt it and regain the session key, which can then be used to make easy secure communications for the leftovers of that session.

While message authentication is useful for effortless or one-way messages, persons that desire to replace multiple messages characteristically set up a secure security context in which to replace multiple messages. A security context is shared along with the communicating parties for the lifetime of a communications session. Once the context and secret have been established (authenticated), Derived Keys can be used for signing and encryption of the messages. Some of features of ws-secureconversation are:

- Similar to SSL
- Create a security context
- Message processing is much faster at both client and Webservice endpoint
- Transparent to the user

E. WS-Federation

A federation is a set of security domains that have established relations for securely sharing resources. A Resource Provider in one security domain can offer authorized access to a resource it manages based on claims about a foremost (such as identity or other unique attributes) that are asserted by an Identity Provider (or any Security Token Service) in another security domains.

The worth of setting up a federation is to make possible the exercise of security principal attributes transversely trust boundaries to found a federation context for that principal. A Relying Person can then utilize this context to grant/deny access to a resource. Establishing a federation context while Identity and Resource Providers activate in different security domains needs agreement between these parties on what claims are required and often requires agreement on methods for firmly conveying those claims more than undefended networks. This offers the foundation for interoperability. In universal it is essential for members in a federation to correspond these requirements over a broad selection of trust and communication topologies. Sustaining dissimilar topologies involves the switch over of metadata recitation endpoint references wherever services may be attained, plus the potential security policies and communication requirements that must be observed when accessing those endpoints. The exchange of this metadata can be further complicated because the participants in a single federation may have different policies and service providers may participate in multiple federations.

A primary objective of WS-Federation is to make simpler the development of federated services during cross-security domains communication and management of Federation Services by reclaim the WS-Trust Security Token Service model and protocol. A selection of Federation Services (e.g. Authentication, Authorization, Attribute and Pseudonym Services) can be developed as variations of the base Security Token Service. Managing, discovering and accessing such services are dramatically simplified when they are all based on a common processing model and speak the same base protocols. Further, reusing an established processing model and having one common protocol reduces the Threat Model variables for implementers and should lead to more robust code.

WS-Federation does not restrict users to a specific security token format. Instead, WS-Federation builds on the WS-Trust encapsulation mechanism, the RST/RSTR, which allows protocol processing to remain agnostic of the type of token being transmitted. This enhances the interoperability and migration of customer deployed products as the industry introduces new and better security token formats.

Building on the WS-Trust foundation the WS-Federation protocol provides mechanisms that simplify interactions between the participants. A well documented method for exchanging federation metadata makes it easy to bootstrap trust relationships and also to determine policies for obtaining services. Cross organizational identity mapping and distributed sign-out improve the utility and overall security of accessing federated service providers by minimizing the user's need to manage many identifiers and tokens. WS-Federation protocol extensions to WS-Trust allow the enablement of advanced services by integrating pseudonym, attribute and claims-based authorization services with generic Security Token Services seamlessly into the basic security model. Pseudonym services and the claims-based authorization model can be used to further satisfy and enhance user privacy requirements across security domains boundaries in a federation. A Resource Provider can describe the set of attributes required to access a resource and an Identity Provider can assert that a particular principal possesses those attributes, without divulging the actual identity of the principal. Finally, within the limitations of standard web browser clients, the security token based capabilities provided by WS-Trust and WS-Federation protocol extensions can be made accessible via HTTP 1.1 mechanisms to be used in browser based scenarios.

F. WS-Authorization

The purpose of WS-Authorization is to describe how access policies for a Web service are specified and eventually

managed. The goal is to describe how claims can be specified within security tokens, and how these claims will be interpreted at the endpoint.

WS-Authorization is designed to be flexible and extensible with respect to both authorization format and authorization language. This enables the widest range of scenarios and ensures the long-term viability of the security framework.

WS-Authorization provides a standard for managing information used for authorization and access policies. As part of this standard, the manner in which claims are represented within security tokens is established. A specification that allows you to specify how the users of your Web service will be authorized against various functionality. When you have a large user base and different users are permitted access to different aspects of your service. This specification (forthcoming) specifies the policy parameters for how Web services will authorize users. Authorization is different from authentication in that once a user is authenticated their authorization will determine which aspects of the service they may access. For example, one may need to authenticate against a service to use that service, but different users will have access to different sets of Web methods on the service. This is determined by their authorization profile, and this specification will provide a schema to uniformly describe that profile.

IV. CONCLUSION

This manuscript concluded with the brief explanation about web services and its security. This survey extends its analysis about the W-Security and its specifications. The specifications of WS-Security is analyzed and explained in detail with its workings and some specification architecture. Security is important for any distributed computing environment. But, security is becoming even more important for Web services due to some reasons. In this survey of security and issues in security is discussed. Web services security is too provided by using WS-Security and also some basic security specifications. Further the security is explained in the way of handling web services security with its specifications. In the future work, the discussion can be about authentication and authorization of users, securing the data of users, and tracking the user activity.

REFERENCES

[1] https://www.facultyresourcecenter.com/curriculum/6534-WS-Authorization.aspx?c1=en-us&c2=0.
[2] http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/ws-secureconversation-1.3-os.html.
[3] http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html .http://fusesource.com/docs/esb/4.4/cxf_security/WsTrust-Intro.html
[4] http://www.networkworld.com/details/6284.html
[5] http://msdn.microsoft.com/en-us/library/bb498017.aspx

Mrs. K. Rajasri received the B.Tech (Information Technology) and M.Tech (Information Security) degrees in computer science and Engineering from Pondicherry Engineering College affiliated to Pondicherry University, Pondicherry. She is Assistant Professor at the Christ College of Engineering And Technology Affiliated To Pondicherry University, Pondicherry She published her manuscript in reputed journals and her research towards on Network security.

Mrs. S. Tamilarasi received B.E in computer science and Engineering From Priyadharshini Engineering College affiliated to Anna University,Vellore and She is currently pursuing Masters in Computer science and Engineering from Christ College Of Engineering And Technology Affiliated To Pondicherry University,Pondicherry. She published her manuscript in reputed journals and interested in Web Security and Network Security and pervasive computing.

Ms. S. Sathiyadevi Received Post Graduation MCA in Computer Science And Applications From Rajiv Gandhi College Of Engineering And Technology Affiliated To Pondicherry University, Pondicherry and She is Currently Pursuing M.Tech In Computer Science And Engineering From Christ College Of Engineering And Technology Affiliated To Pondicherry University,Pondicherry. She published her manuscript in reputed journals and interested in Web Security and Network Security.