WSS - Web Service Security

Pieter de Boer
Bart Dorlandt
Wouter Borremans

December 16th, 2004

1 What is WSS?

The WSS Techical Comittee developed the OASIS[2] specification. This specification describes enhancements to SOAP to provide message integrity and confidentially. The mechanisms can be used to accommodate a variety of security models and encryption technologies. The specification refers to a set of extentions and modules as the 'Web Service Security: SOAP Message Security' or 'WSS: SOAP Message Security', it is very flexible and is designed to be used with Kerberos, SSL and security models such as PKI. This specification supports multiple security token formats, multiple trust domains, multiple signature formats and multiple encryption technologies.

The mechanisms described above do not offer complete security solutions for web services. Instead, the specification is a building block That can be used in conjuction with other web service extentions and higher level application specific protocols.

2 Notation and Terminology

This section describes the notations, namespaces and terminology in the OASIS specification.

2.1 Notational Conventions

When describing concrete XML schema's, this specification uses a convention where each member of an element or attribute is described using a XPath-like notation. (e.g. /x:MyHeader/x;SomeProperty/@value1) The use of any indicates the presense of an element wildcard (<xs:Any/>). The use of @Any indicates the presence of an <xs:Anyattribute/> element. Readers are persumed to be familiar with the terms in the Internet Security Glossary.

2.2 Namespaces

The XML URI's that *MUST* be used implementing this specification are as follows:

```
http://www.docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-wssecurity-secext-1.0.xsd
http://www-docs.oasis-open.org/wss/2004/01/
oasis-200401-wss-wssecurity-utility-1.0.xsd
```

This specification is designed to work with the general SOAP message structure and message processing model.

3 WSS in Microsoft .NET framework

WS-Security has been implemented by Microsoft in the .NET[1] platform under the hood of Web services Enhancements. Web services Enhancements is Microsoft's API to support advanced Web services functionalities, like security policies. WS-Security can be handled by this API.

3.1 Adding securitytokens

To be able to use WSS, the SoapContext used for a request message should have a security token assigned. A UsernameToken can be added to the SoapContext to do just that. This is done as follows:

```
// Instantiate the HelloWorldWse class.
HelloWorldWse myService = new HelloWorldWse();

// Get the SoapContext object for the
// outgoing message.
SoapContext myContext = myService.RequestSoapContext;

// Instantiate a new UsernameToken object.
UsernameToken myToken = new UsernameToken(myName,
  myPwd, PasswordOption.SendNone);

// Add the token to the SoapContext.
myContext.Security.Tokens.Add(myToken);

// Generate a signature using the username token,
// and add the signature to the SoapContext.
myContext.Security.Elements.Add(
  new Signature(myToken));
```

The WSE-API will now create a *UsernameToken* element in the message's *Security* header. In this case, the password itself is not sent in the request, because the signature can be used for authentication.

3.2 Authenticating security tokens

Authenticating the sent security token will be done automatically by WSE in case of Kerberos and X.509-based tokens. When authentication needs to be done on Windows accounts, passwords need to be sent in plain-text and .NET needs to run with full system privileges. A better way to authenticate based on username/password combinations, is to register a usernametoken manager with WSE. WSE can then simply compare the password sent over SOAP with a password you fetch from a database, textfile, or whatever you wish. If a signature is used, WSE can check that too, based on the given password.

4 WSS & Java

In this chapter we'll see how the client and server interpretation of the WSS messages are handled. The information we used of this site Web Services and Securtity and Java. [6]

4.1 Server Side

The server application receives a message with a number of security tokens. Web Services Security for Java (WSS4J) extracts the tokens from the WSS message and returns them to the server application.

The encrypted portions in the message are decrypted with a secret/private/Kerberos key. The server can only decrypt message targeted for itself.

On the other hand the server can also use WSS4J to provide a list of all XMLDS signatures in the WSS message. After WSS4J has provided a list of all signatures, the server application can ask WSS4J to process any individual signature. This is also possible with a list of encrypted elements.

4.2 Client Side

The client use the WSS4J to attach security tokens to the SOAP message. WSS4J will author the complete WSS compliant XML structure of the security token, wrap the security token inside a WSS header, and return the completed WSS message back to the client application.

If a client application wants to sign a portion of a SOAP message using a private key with an X509 certificate it uses the SOAP message that already has the certificate. This message uses WSS4J to sign a particular portion of the WSS message using the private key associated with the certificate. The

selection of the particular portion is based on the fragment identifier [3] or the XPath expression [4].

The message will be generated as follows:
A client application wants to encrypt a particular portion of a SOAP message. The client application first adds the certificate of the intended recipient of the message as a WSS token. It then provides the fragment identifier or an XPath expression to WSS4J and asks the application to produce XML encrypted version of the message. WSS4J produces the encrypted portion of the message and returns the completed message back to the requesting client application.

For an example of a WSS message that contains two tokens can be viewed at the site of hitbox [5]

5 Conclusion

WSS does not offer complete security solutions for web services. Instead, the specification is a building block That can be used in conjuction with other web service extentions and higher level application specific protocols.

References

[1] Securing .NET webservices, Jeannine Hall Gailey , `http://www.devx.com/dotnet/Article/19986/0/page/4`

[2] OASIS, WSS Techinal Comittee, `http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf`

[3] Fragment identifier - `http://schemas.xmlsoap.org/ws/2003/06/utility`

[4] XPath - `http://webservices.xml.com/pub/a/ws/2003/10/28/jwss.html#resources`

[5] WSS - `http://webservices.xml.com/2003/10/28/examples/Listing1.html`

[6] Web Security for Java - `http://webservices.xml.com/pub/a/ws/2003/10/28/jwss.html`