A Taxonomy of Authentication Techniques for Web Services

A. Jesudoss [#1], N. P. Subramaniam[2]

[1]*Research Scholar, Sathyabama University*
Chennai, India
[2]*Asst. Professor, Department of EEE, Pondicherry Engineering College*
Puducherry, India

Abstract

Authentication is one in all the under-addressed problems within the field of Web Services. It is the process of making certain that the one who tries to access the service is basically the individual that he claims to be. It enables to ascertain trust between parties engaged in business transactions or any service. It also enables the users to prove user's identity on online transactions. Web service authentication is not like Web site authentication because the authentication process is made by another Web service. It is about one service interacting with another service rather than service interacting with another service. In this paper, the merits and demerits of various authentication mechanisms prevailing in e-commerce world are discussed. A comparative study on the existing techniques is also clearly shown.

1. Introduction

All Web services refer to the technology that allows accessing services across the Web. Web services are modular applications. Web services are used for various business functions ranging from simple to complex tasks. Passwords are used as a tool to authenticate and authorize the identity requesting to access the resource.[1] The important factor about authentication is how to make the password more secured and at the same time maintain acceptable usability also. [2]

Business people and enterprises need to know information about people and various aspects for conducting business effectively. Hence, it is very crucial to ensure that the information shared among them is reliable and secured. While consuming a Web Service or while going for online transaction, one must be sure about the identity of the person with whom they are interacting and from whom the data received should be trustworthy. The Web services security architecture[3] is given below in figure 1.

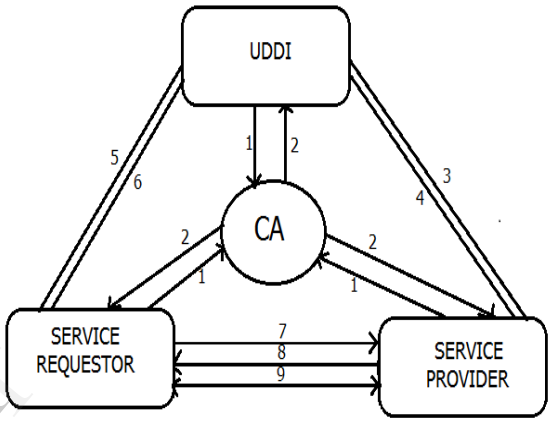

Figure 1. Security Architecture for Web Services

The security architecture of Web services consists of the following steps[4]:

Step 1: The entities service provider, service requestor and UDDI must obtain a valid certificate from the Certification Authority (CA) for proving their identity.

Step 2: CA verifies their proof of identity and issues certificate to them.

Step 3: Service Provider registers the Web service with UDDI.

Step 4: UDDI accepts the registration and acknowledges the service provider.

Step 5: Service Requestor requests the UDDI for desired service.

Step 6: UDDI provides WSDL to the Service Requestor describing the services and its location in the WSDL file[5]

Step 7: With the help of WSDL and valid credentials, Service Requestor requests the desired service to the Service Provider.

Step 8: Service Provider validates the user credentials and on successful verification positive acknowledgment is sent to the Service Requestor. Otherwise, negative acknowledgment will be sent.

Step 9: Connection is established. Service Requestor consumes the service from the Service Provider.

2. Emerging Authentication Mechanisms

There are various ways to authenticate a Web service. Some of the emerging techniques are discussed below:

2.1 Emerging Authentication Mechanisms

The Web service can be authenticated using underlying operating system's infrastructure such as Active Directory. When Windows operating system is used, IIS Web server can be used to authenticate a Web service. The IIS refers to the Active Directory for verifying the user credentials supplied by the user. The IIS Web server verifies the user credentials against the entries in Active Directory of Windows Server. If the credentials provides match with the one in the Active Directory, then provides access to the protected web service. The granted access will also be based on the privileges set for the user on the Active Directory. IIS can also provide anonymous access to the Web service. The IIS should be secured perfectly in order to avoid intruders exploiting the vulnerabilities of the Web server. The main drawback of Windows-based security is that it can be used only by the Windows users. Since IIS is Microsoft's product, it extends service only for the Windows users.

Once the Web server is configured with Windows-based authentication, then all calls to the web service will be interrogated by the Web server. There are four types of windows-based authentication such as BASIC, DIGEST, INTEGRATED and CUSTOM authentication. The different authentication method uses different transport protocols.

2.1.1 Basic Windows Authentication

Basic authentication is a typical authentication protocol defined by RFC 1945 and followed by major industries. The user sends credentials such as username, password and the domain name to the server. The server compares the user credentials against the database which contains the list of legitimate users. The service will be denied by the server if the user does not include appropriate security credentials. The IIS will reply with HTTP error 401. When Basic Authentication is used, the browser will prompt the user to enter network. The security credentials can be provided programmatically by the client application. Basic authentication is used as the default authentication if no authentication is specified explicitly.

2.1.2 Digest Windows Authentication

Digest authentication avoids sending password in clear text over HTTP for security reasons and hence it hashes the user credentials using hashing algorithm such as MD5. The server calculates the expected digest and compares it with the received hash. If it matches, access will be granted else it will be rejected. HTTPS is not needed for secured transmission. IIS Web server supports Digest authentication. Digest Authentication is standard used by major industries and it is defined RFC 2617. However, it requires Active Directory to store the password. Without Active Directory, you cannot configure the Web server to make use of Digest Authentication.

2.1.3 Integrated Windows Authentication

Integrated authentication is another authentication method that uses a proprietary hashing algorithm. The drawback in this method is that both the client and the Web service runs on Windows platform. Integrated authentication avoids sending user credentials in plain text. It check the client's identity such as security token for validating the client. If the client's identity is not adequate and you need to explicitly provide other details about the client. IIS supports Integrated Windows Authentication.

2.1.4 Custom Authentication

Vulnerabilities may exist on the Web Server or on the Operating System itself. Hence, an application must on rely on the underlying infrastructures alone. Such vulnerabilities may invite the hackers and intruders. In order to avoid these problems, custom authentication must be preferred. The user credentials are handled differently for different types of custom authentication.

If the service requestors provide enough proof of their identity, then they will allowed to invoke the Web service method. Otherwise, an exception will be thrown. Custom authentication must be preferred when the service consumers do not have accounts on the server. A custom authentication provides absolute

control in accessing the Web service. In custom authentication using IIS, anonymous access check box must be selected and Authentication mode must be set to "NONE" for providing anonymous access to the Web service.

In custom authentication, there are variety of authentication techniques for Web services such as log-in method, SOAP headers, Cookies and SOAP headers, SOAP extensions, Encrypted SOAP extensions, etc. Each of the technique is unique and can be altered according to the requirements of the service provider. The custom authentication techniques are discussed as follows.

a) Log-in Method

In log-in method, the username and password are sent in plain text. Hence, custom authentication must be accompanied by HTTPS. The session variables are used for storing the user credentials. The value of the credentials remains in the session variable throughout the session. It enables to identify whether the user requesting for the service is legitimate user or not. Since session variable is used for authentication, it is very essential that it should be encrypted.

b) SOAP Headers

The SOAP header information is appended to the SOAP payload before transmission. On the Web service end, it is separated from the SOAP payload and verified. Once the user credentials obtained from SOAP header is verified successfully, then it provides access to the Web service. This can be achieved even without calling the Web service. The SOAP header attached to the SOAP message is transmitted as a clear text. Hence, a secured communication channel such as SSL is essential for transmitting SOAP headers. Otherwise, the SOAP message will be intercepted by the hackers easily.

The System.Web.Services.Protocols is the .NET namespace that provides support for SOAP headers. First you have to create a class that is derived from SoapHeader class and then user name and password must be created as members of the class. The SOAP header class members must be public and in the form of fields or data members only. The service provider generates the Web Services Description Language (WSDL) document associated with the Web service. It contains information about the SOAP header and other details essentials for the clients.

SOAP header variable along with the SoapHeader attribute can be used with any Web method as follows.

```
[SoapHeader("AuthHeader")]
[WebMethod]
public int Add(int n1,int n2){...}
```

AuthHeader given by client will be initialized by .NET automatically. Now the Web method has to authenticate the caller in the sensitive methods. This is done using the Authenticate() method, which obtains the credentials from the member variables of the SOAP header.

When the .NET client includes a Web reference to the Web service using SOAP header, derived class will be generated by the .NET with public variables only. .NET will also add to the wrapper class a matching member variable.

The headers will be passed to the Web service and it is used to authenticate the caller. If the credentials are static information, then the SOAP header initialization can be encapsulated by client in the wrapper class constructor.

c) SOAP Extensions

All calls to Web service can be intercepted and all pre- and post-call processing using SOAP Extensions can be performed. The Web service client can also deal with the SOAP extension. For example, SOAP extensions can be used to compress the SOAP payload and to encrypt the information. SOAP extensions are used for payload encryption which results in amazing challenges such as the distribution of keys, etc. It is an inevitable fact that SOAP extensions are more reliable than SSL, particularly for a public service.

SOAP extensions can be used for sending the user credentials. Secured channels are not essential when the the payload is encrypted. Hence, the developers can utilize SOAP extensions for communicating with SOAP messages. SOAP extensions are used to intercept the SAOP message and user's code can be inserted into the SOAP message. This extends the capability of the SOAP message. The SOAP extensions can be used for security, transaction management, packet forwarding, identifying path, etc.

SOAP headers provide perfect mode of transport for sending the user credentials. SOAP extensions are often used for scrutinizing the SOAP headers and it also denies the client fails to prove its identity. Hence It is prudent to combine the two and develop secured Web services that demarcates business logic from authentication aspect of security.

3. A Comparative Study on various Authentication Techniques

In this paper, three Windows-based techniques and five custom authentication techniques have been examined carefully. There are many combinations and variations on this theme. The table 1 shows the various authentication methods. Various factors are to be considered before choosing an authentication method. First, encryption factor is to be considered whether the user credentials need to be encrypted or not. If the credentials are transmitted as a clear text, the need for secured channel such as HTTPS is mandatory. Secondly, Operating system of the client and Web service server should be considered. Are they same or different? Does the authentication system provide cross platform support. Thirdly, the platform requirement on the server side and the client side are to be considered. Scalability is another important factor and ensures whether authentication system serves the growing needs of the client. Fourthly, authentication system provides single sign-on (SSO) feature[6] or not. Is it necessary to authenticate only the first call to the web service or all the calls have to be authenticated. Does it maintain any session or not. Finally, authentication system should be tested whether it suffers from low bandwidth due to any of the factors mentioned above. When the bandwidth is affected, we must consider what problem affects the bandwidth and how it can be rectified should be considered. When administrators create policies for passwords, they increase burden for the users in managing the passwords.[7] It is always an interesting challenge for the administrators to handle logging for distributed environments particularly the Web service.[8]

Windows-based authentication can be chosen as it alleviates the manual work such as writing a code for authentication or performing a procedure for authentication for deploying a secured web service. In custom authentication, it should be done manually and the developer should define the entire authentication process. Among custom authentication, Log-in method can be the best choice. Because it is very easy to deploy and provides declarative security for the Web service. It is easy to implement. Implementing SOAP header or SOAP extension is difficult than just invoking a method, especially for legacy client applications such as Visual Basic 6.0. There is no support for SOAP headers or extensions in legacy application softwares. We would strongly suggest that an authentication system must combine both Windows Authentication and Custom Authentication techniques to render better security

4. Results

The experimental results shown here demonstrate the basic difference between HTTP Basic Authentication and Form-Based Authentication. In HTTP Basic Authentication, windows popup dialog box will prompt for username and password.

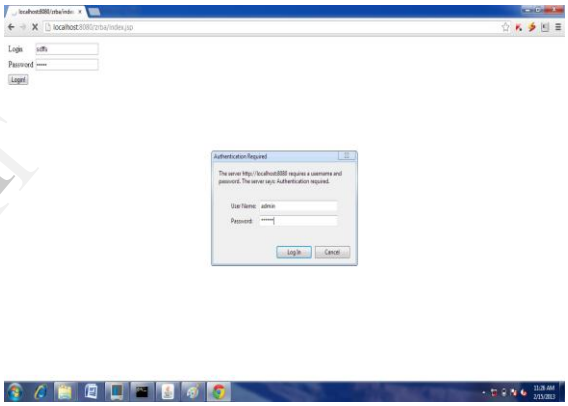

Figure 2. HTTP Basic Authentication

In Form-based authentication, customized form can be used as a login screen. The developers are free to design their own form for authentication purpose.

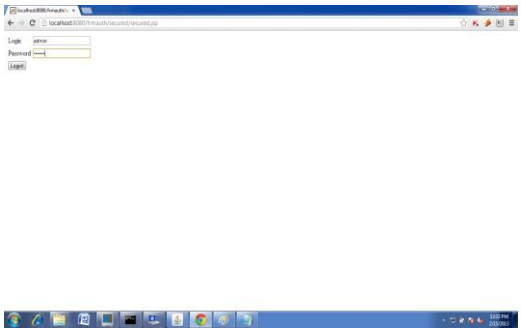

Figure 3. Form-Based Authentication

guides is available as both Word and PDF files as <format.doc> and <format.pdf>. It contains lines and

boxes showing the margins and print areas. If you hold it and your printed page up to the light, you can easily check your margins to see if your print area fits within the space allowed.

5. Conclusion

This paper helps to understand various authentication techniques for the Web service security. It focuses on the authentication aspect for windows platform and IIS. The investigation of authentication in this paper is limited to homogenous environment such as Windows and does not support heterogeneous platforms. The identity management must be integrated into browsers, operating systems, web servers such as IIS, applications, etc[9]. The ultimate aim of this paper is accomplished by examining coherent authentication techniques. The examination of various authentication techniques leads to the development of an efficient and the best authentication model for securing Web services. This paper enables to develop a novel authentication framework which combines both the Windows-based IIS authentication and custom authentication and provides greater security to the field of Web service.

References

[1] Markus Jakobson, Richard Chow, Jesus Molina, "Authentication – Are we doing well enough?", IEEE Security & Privacy, pp. 19-21, February 2012.
[2] Heather Crawford, "Applying Usable Security Principles to Authentication", ACM 2011.
[3] Zhiyi Qu, Yuanting Ge, Kaiyuan Jiang, Tiangang Lu, "Key Issues in Building Web-based Services" , in the proceedings of Third International Conference on Next Generation Web Service Practices, IEEE 2007, pp. 119-122.
[4] Yuan Rao, Bo-Qin Feng, Jin-Cang Han, Zun-Chao Li, "SX-RSRPM: A Security Integrated model for Web services", in the proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004, pp 2953 – 2958.
[5] Jie Xu, Erica Y. Yang, and Keith H. Bennett, "A practical Approach to Secure Web Services", in the proceedings of the ninth IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing", IEEE Computer Society.
[6] Eve Maler, Drummond Reed, "The Venn of Identity – Options and Issues in Federated Identity Management", IEEE Security & Privacy, pp. 16-23. April 2008.
[7] Stephen Farrel, "Password Policy Purgatory", IEEE Internet Computing, pp. 84-87, October 2008.
[8] John Steven, Gunner Peterson, Deborah A. Frincke, "Logging in the Age of Web Services", IEEE Security & Privacy, pp. 82-85, June 2009.
[9] Rachna Damija, Lisa Dusseault, "The Seven Flaws of Identity Management", IEEE Security & Privacy, pp. 24-29, April 2008.