A Scalable Web Service security model

Khalil Challita[1], Hikmat Farhat[1], Joseph Zalaket[2]
[1] Notre-Dame University
[2] Holy Spirit University of Kaslik
Computer Science Department, Lebanon
{kchallita,hfarhat@ndu.edu.lb, josephzalaket@usek.edu.lb}

ABSTRACT: *Through the proposed research we have presented a design of a security model that encapsulates the basic modules needed for securing the access to a web service, which are authentication and authorization. We also ensure that the proposed model relies on WS-Security standards and another application layer technology, namely the "Lightweight Directory Access Protocol". In the work, we in addition, used and tested it, and provided several test case scenarios. Besides, we have conducted an evaluation in terms of performance which is carried out in order to reduce the concerns about security bottleneck and overheads. Finally, we highlighted the benefits of the proposed design with empirial results.*

Key words: Security model, Web Services, SOAP, WS-Security, WSS4J, LDAP

Received: 28 October 2011, Revised 30 November 2010, Accepted 5 December 2010

1. Introduction

Today's companies are more intrigued to publish web services as a scheme to generate additional revenues, as they are selling their existing functionalities over the Internet and making use of the low cost communication protocol *Simple Object Access Protocol* (SOAP). As mentioned in [5], the fast increasing number of web services is transforming the web from a data oriented repository to a service oriented repository.

It is important to say that given the reality of today's open networks, it is just impossible to conduct business transactions in a networked environment without full security capabilities. Although web services are a boon to e-commerce [19], they come at a high cost of shaky security. Current methods and technology cannot fully ensure trustworthiness in loosely coupled web services. Current interests remain focused on implementing web services where the lack of security is not a bottleneck to industry's adoption of this Internet model. So far, significant progress has been made towards making the web service technology a suitable solution for areas such as e-business and e-government and e-commerce [18]. However, there are still some issues blocking the progress of the wide scale deployment of web services, one of those main issues is the security of web services.

Our aim in this paper is to provide a combined security model for web services that ensures both authentication and authorization, in order to allow a client and a service to communicate securely, and be protected from potential problems such as unidentified client requests or unauthorized access to resources. Our model combines components that belong to different perspectives of security technologies such as *Web Service Security* (WS-Security) standards, application level protocol, and application layer processing. Our model is inspired from the one provided by Garcia and Toledo [8], where only the issues related to the security of a message, namely confidentiality and integrity, are dealt with. We complement the work done in [8] by providing a combined security model that covers both authentication and authorization based on predefined web service security standards such as WS-Security that defines standardized syntax dictating security goals that can be carried out within the messages exchanged between a client and a service. Our proposed model tends to provide a security mechanism

that, once applied, will indeed achieve the issues of identity verification and access control management.

The rest of this paper is divided as follows. We introduce in Section 2 the major concepts and technologies in web services' security, in addition to widely known projects built to support web services' security. The related work is presented in Section 3. We describe in Section 4 our model that ensures both authentication and authorization. The implementation of our model is given in Section 5. Finally, a benchmark discussion is given in Section 6.

2. Web services' security

Lafon [12] defines web services as programmatic interfaces made available over the World WideWeb to enable application-to-application communications. They enable software applications to communicate and conduct transactions by adopting the *Extensible Markup Language* (XML) as a data exchange format and industry standard delivery protocols, regardless of the programming languages they are developed with and the platforms they are deployed on. Yu et al. [18] state that web services support direct interaction with other software agents using XML-based messages exchanged via Internet based protocol. Examples of web services include online reservation, stock trading, auction, etc. Web services are currently being widely adopted as a platform independent middleware. However, web services were not that interesting until a few years ago. Thanks to the major IT development the last few years, most people and companies have broadband connection and are using the web increasingly.

2.1 The need of a web service security standard

Security is an important factor for deploying web services, as Yu et al. [18] mention. Web services need to be concerned with the security aspects of authentication, authorization, confidentiality, and integrity. According to Tang et al. [16], SOAP, which is a messaging protocol that allows applications to exchange information, does not provide security for messages, since it brings threats to both sender and receiver of the message. That is why the web wervice security specification was developed. Nadanlin et al. [13] define web service security (WS- security) as a set of communications protocols set to address security concerns. The WS-Security specification describes enhancements for the SOAP messaging to achieve message integrity, confidentiality, and authentication. Developed by a committee in Oasis-Open, it specifies mechanisms that can be used to accommodate a wide variety of security models and encryption technologies. It is °exible and can be used within a variety of security models like Secure Socket Layers (SSL) and Kerberos.

The web services' security challenge specified by Hondo et al. [10] is to understand and assess the risk involved based on an existing security technology, and at the same time follow the standards and understand how they will be used to offset the risk in new web services. Most of the current security discussions address identity authentication and message exchange privacy. Additional security measure at the application level could be of use, targeted at preventing authorized visitors from performing unauthorized actions. In this paper, our main concern is to address the following two fundamental security issues: authentication and authorization.

2.2 WS-security specifications

The WS-Security specification described in [13] provides mechanisms to address the three security requirements: authentication, integrity and confidentiality. With WS-Security, we can selectively employ one or more mechanism to implement a specific security requirement.

The specification given by Zhang [19] provides a mechanism to associate security tokens with message contents. It is designed to be extensible and supports multiple security token formats. The mechanisms provided by this specification allow to send security tokens (that are embedded within the message itself) in order to achieve message integrity and message confidentiality. Note that the WS-Security standard given in [13] uses the XML Encryption standard to allow encryption of any portion of the SOAP message.

2.3 Some WS-security projects

We next present several web services' security projects on which we rely to implement our model in Section 5. Then we describe the LDAP protocol and the Active Directory, which constitute a main component of our model, because they are at the basis of the authentication process we aim to achieve. WSS4J Apache WSS4J is an implementation of the WS-security [5]. It is an open source java library that can be used to sign and verify SOAP messages. It also provides resources for building interoperable, trusted web services using the WS-security standard. The libraries can be deployed to provide

protocol support for both client and server applications.

We are interested in using WSS4J as a library within our model because it is an open source project, and because it is interoperable with "Java API for XML based Remote Procedure Calls" and .NET server/clients.

SOAP Box et al. [5] define SOAP as a simple XML-based protocol to let applications exchange information over an Application Layer protocols like SMTP, HTTP, or HTTPS. The SOAP specification is currently maintained by the XML Protocol Working Group of the World Wide Web Consortium. SOAP can encapsulate WS-security information. As described by O'Neill [14], security data in SOAP include security tokens to indicate the identity of the sender, and a digital signature to ensure that the SOAP message has not been modified since its original signing. SOAP is used to send data from one application to another. It is supported by all Internet browsers and servers, and at the same time allows applications to communicate when running on different operating systems with different technologies and programming languages.

Axis The Apache organization [1] created Axis2 to be a core engine for web services. Axis2 not only provides the capability to add web services interfaces to web applications, but can also function as a stand-alone server application.

We used Axis2 to create an application based on our model (given in Section 5), and where both the client and the server were developed as stand-alone applications, without having to create a separate web application to act as a service.

Rampart Apache Rampart [3] is a project that implements the WS-Security standard for the Axis2 web services engine created by the Apache Software Foundation. It provides some security features to web services.

Lightweight Directory Access Protocol According to Koutsonikola et al. [11], Lightweight Directory Access Protocol (LDAP) is an application layer protocolfor querying and modifying directory services running over TCP/IP. We use LDAP to authenticate users requesting a service against a Service provider's active directory in order to restrict accesses to known and specified users only. Moreover, we chose LDAP because it is widely supported, very general and in- cludes basic security, and it can support many types of applications.

Active Directory Active Directory (AD) [6] is a technology created by Microsoft providing multiple network services such as LDAP directory services, Kerberos based authentication and DNS base naming and network information. It is a type of databases that can be searched to provide useful network infor- mation. We chose Active directory in conjunction with LDAP because it allows users and applications to make use of published information over a network without requiring any knowledge about this network. Moreover, an AD is an optimized database for querying which makes information retrieval easier and faster.

We next give an overview of some researchers' work done in the field of web services security, and summarize their main contributions to the field.

3. Review of Related Work

Yamaguchi et al. [17] proposed an application programming model called "Web Service Security Application Programming Interfaces" to simplify the programming for end users who are not very familiar with WS-Security. Their model was based on the Service Oriented Architecture (SOA), the WS-security requirements, and on the existing APIs proposed by Microsoft. It consisted of six APIs that tend to achieve confidentiality and integrity through signatures and encryption.

Bhargavan et al. [4] addressed the problem of securing sequences of SOAP messages exchanged between web services providers and clients. Their work confirmed the inefficiency of using WS-Security alone for each message and that the integrity of a whole session as well as each message should be secured. They relied on WS-Secure conversation, which goal is to secure sessions between two parties, and on the WS-Trust, which describes how security contexts are obtained.

Gutierrez et al. [9] intended to describe a security model for web services in order to facilitate the development phase. Their model is based on web service security requirement specification, web service security architecture design and web service security standard selection. The research focused mainly on the web services security architecture.

Rahaman et al. [15] describe web services security architectures in a simplified way using WS standards, in addition to

addressing the issue of attacking a SOAP message from XML rewriting attack. The research focused on message level security and discussed two different message flows that use (or do not use) SOAP message structure information.

Felix et al. [7] addressed the scalability and flexibility limitations of the WS authentication model where the acquirement of identity claims requires online interactions with security token services, thus introducing communication overhead and creating performance bottlenecks. They presented a new model where they addressed these limitations through two concepts: credentials for claim inference and claim-based issuer references. They showed how credentials are used both to increase the scalability and to reduce the number of online token requests. They also showed how the simultaneous usage of security tokens and credentials results in several advantages when compared to credentials used in trust management models.

Zhang [19] enumerated the main challenges that threaten a web service and make it "untrustworthy". He proposed a solution to address these challenges by adding an additional layer (i.e. WS-Trustworthy layer) on top of the WS-Security layer.

Garcia and Toledo [8] proposed a security model for web services that is based on semantic security policies. Their main goal was to ensure confidentiality and integrity of a SOAP message. The main components of the security model they designed are equivalent to some XML elements of the WS-Security, XML encryption and XML Signature standards. Note that they only addressed the issues of confidentiality and integrity.

To our knowledge, no single model addressed both authentication and authorization at the same time. We next give the description of our model that addresses these two security issues.

4. Authentication and authorization-based model

Our model, given in Figure 1, is inspired by Garcia and Toledo's one [8], and can be considered as a complement to their model since they focused only on the security and integrity of a message, regardless of the identity of the message issuer and her access rights. Our main goal is given in the main module, namely "Securing Web Services", which in turn is composed of two submodules: Authentication and Authorization.

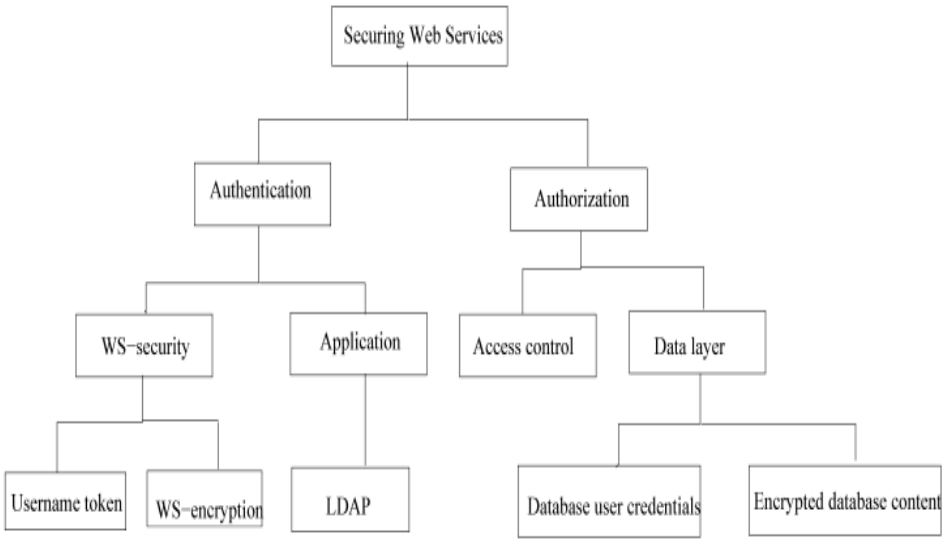

Figure 1. Authentication and Authorization Diagram

4.1 Authentication

Authentication is achieved through the use of WS-Security standard and, more specifically, the "UsernameToken" that enforces the use of a username and a password. Note that both the username and the password will be protected inside the SOAP message through encryption. In addition, an application-level protocol (i.e. LDAP) is integrated in the authentication mechanism in order to ensure that access is only allowed to known and identified users.

4.2 Authorization

Authorization is achieved through two layers:

1. The “Access Control layer”, whose purpose is to identify a requestor and to check her access rights;

2. The “Data Layer”, which is related to a Database system containing the logical mapping of users to the requested services for access control purposes. The data layer is composed of two submodules: the “Database user credentials” that are used to retrieve the required information from the Database; and the “Encrypted database content” that stores encrypted data.

5. Implementation of our model

In this section we describe the application that we built on top of our proposed model in order to test its feasibility, e±ciency and performance.

The diagram in Figure 2 shows the engine structure of the created application.

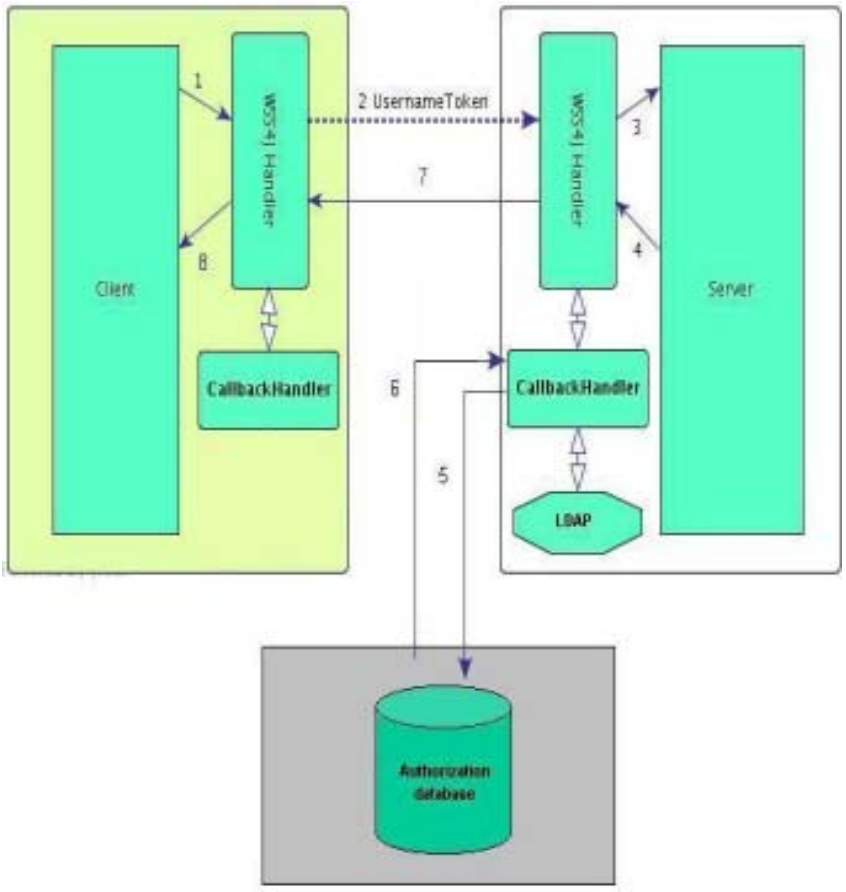

Figure 2. Application Engine Diagram

The application engine we created is divided into two main components: the client and the server. The client is the part that requests a service providing the service the necessary credentials, the server is the service provider, its goal is to authenticate the requestor and validate its access rights before sending back a response. Our engine is based on WSS4J, which is an Apache project implementing WS-Security specification. It also uses Rampart engine, which is another Apache project based on WSS4J to secure SOAP messages.

We next explain the steps shown in Figure 2.

1. The client issues a service request. This request is intercepted by the WSS4J handler, which then invokes the client CallbackHandler class where the user password is provided.

2. The client’s request now holds the UsernameToken with the corresponding credentials and passes it on to the server.

3. At server side, the WSS4J handler intercepts the message containing the token.

4. The server then invokes its corresponding Callbackhandler class where the credentials are retrieved in order to be processed.

5. The credentials are passed through LDAP to the corresponding server's Active Directory where they are validated for authentication.

6. In case of successful authentication, the user information is sent to the database for authorization. The role of the user is fetched as well as its corresponding services. Once the allowed services are retrieved they are compared to the incoming service name call and the validation result will be communicated in the next step back to the client.

7. This is the result (of success or failure) that is communicated back to the client through its CallbackHandler.

8. The response is sent to the client application either in an exception mode when the authorization failed, or as a service result in case the authorization succeeded.

6. Benchmark

The aim here is to examine the performance level of our model. The next scenarios were performed using the benchmark tool Mercury LoadRunner 8.1.

6.1 Scenarios

Scenario 1: Web Services with UsernameToken and LDAP In this scenario our application uses a UsernameToken and performs an LDAP search in the active directory. The results collected reflect how the combination of both WS-security and application level protocol affect the overall performance and response time of the system. The results in Figure 3 show the response time in milliseconds by number of simultaneous users access.

Number of users	Average transaction response time
10	235.46
25	760.78
50	1914.75
100	3853.74
150	5238.35
200	9351.27

Figure 3. Application Engine Diagram

As we notice, the average response time per user when having a full load of 200 simultaneous users is almost 46.7 ms per user.

Scenario 2: Web Services with SSL

For SSL, and as we can see from Figure 4, the average response time when having a full load of 200 simultaneous users is around 21.1 ms per user. Even though the results show a clear difference in response time when using SSL against WS-Security and LDAP, however, there are many reasons that weaken the value of SSL as we explain in the next subsection.

6.2 Advantages and limitations

The main advantage of our authentication process is that it is a direct authentication process, which means that the client and the web service trust each other to securely handle the authentication credentials.

The WS-Security standard and the use of LDAP play an important role in our model because they have some advantages over other authentication techniques.

For example, the main limitations of SSL are:

1. SSL is only good for POINT-TO-POINT communication.
2. Authentication becomes difficult when using SSL.
3. SSL is bound to HTTP protocol.

4. Encrypting or signing part of a message is not possible with SSL.

Number of users	Average transaction response time
10	98.89
25	323.33
50	825.25
100	1676.38
150	2310.11
200	4226.68

Figure 4. Web services with SSL benchmark

LDAP is widely supported, very general and includes basic security, and it can support many types of applications. It also allows the reuse of domain credentials so one does not necessarily have to create a separate database to list the users who are entitled to call services.

As for the authorization process, the main advantage is that it is an application level process and thus it could be easily customized according to each service provider. It is a scalable module that offers easy integration of new rules and new entities that make it ready for any change of service policy.

7. Conclusion

In this research we addressed and proposed a solution to the problem of authentication and authorization in web services security. For that purpose we created a model that combines security technologies from multiple ends. First, we used the WS-Security standards, mainly the security token "UsernameToken" in the authentication process; next we combined this standard with the use of an application level protocol, which is the LDAP for credentials validation. Then in the authorization process, we relied on an application level manipulation, where we created a logical and physical model to underline the mappings between users and their corresponding business roles, as well as between the roles with their related services. These mappings are at the basis of the authorization control management since they decide the access rights for a user requesting a service. Moreover we implemented our model and conducted different test case scenarios to put under examination its efficiency. We also conducted benchmark scenarios to evaluate our model's performance in terms of response time and compared it to other security techniques.

Even though our model deals with issues of authentication and authorization, it needs to be extended in order to give a wider coverage of security, namely confidentiality and integrity.

Fortunately, our model is flexible enough to encapsulate new security measures whether at the level of WS-security (since we can add new tokens or even use different standards like WS-policy or WS-Trust), or at the application layer by integrating new entities to the data model.

References

[1] Apache axis2/java next generation web services. (2008). *In:* http://ws.apache.org/axis2/.

[2] The apache software foundation. (2008). *In:* http://ws.apache.org/wss4j/.

[3] Rampart:(2008). Ws-security module foraxis2. http://ws.apache.org/axis2/modules/ ram- part/ 1 3/security-module.html,

[4] Bhargavan, Corin, Fournet, Gordon. (2007). Secure sessions for web services. *ACM Transactions on Information and System Security*, .

[5] Box, Enhebuske, and Kakivaya. Simple object access protocol. (2000)http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[6] Dias, J. (2002). A guide to microsoft active directory (ad) design. *Department of Energy Lawrence Livermore National Laboratory* .

[7] Felix, Pedro, Ribeiro (2007). A scalable and flexible web services authentication model. *Proceedings of the 2007 ACM workshop on Secure web services*, p 62-72.

[8] Garcia, D,. and de Toledo, F., (2008). Web service security management using semantic web techniques, *In*: *CM symposium on applied computing*, p. 2256-2260.

[9] Gutierrez, Fernandez-Medina, and Piattini. (2005).Web services enterprise security ar- chitecture: a case study. *Proceedings of the 2005 workshop on Secure web services*, p 10-19.

[10] Hondo, Nagaratnam, and Nadalin (2002). New developments in web services and e- commerce, securing web services. *IBM Systems Journal*, 41.

[11] Koutsonikola, V., Vakali, A. (2004). Ldap: Framework, practices, and trends. *IEEE Internet Computing*, 8 (5) 66-72.

[12] Y. Lafon. Web services activity statement. http://www.w3.org/2002/ws/Activity, (2008).

[13] Nadalin, A. (2006). Web services security: Soap message security 1.1. http://docs.oasis- open.org/wss/v1.1.

[14] O'Neill, M. (2005).Web services sercurity. *McGraw-Hill*.

[15] Rahaman, Schaad, d Rits (2006). Towards secure soap message exchange in a soa. *Proceedings of the 3rd ACM workshop on Secure web services*, p.77-84.

[16] Tang, Chen, Levy, Zic, Yan (2006). A performance evaluation of web services security, *IEEE Computer Society*, p. 67-74,

[17] Yamaguchi, Chung, Teraguchi, and Uramoto (2007). Easy-to-use programming model for web services security. *Proceedings of the The 2nd IEEE Asia-Pacific Service Computing Conference*, p. 275-282.

[18] Yu, Liu, (2008) Bouguettaya, and Medjahed. Deployingand managing web services: issues, solutions, and directions. p 537-572. VLDB.

[19] Zhang, J., (2005). Trustworthy web services: actions for now. *In: IT professional*, p. 32-36.