A Performance Evaluation of Security Mechanisms for Web services

Bachar Alrouh and Gheorghita Ghinea
School of Information Systems, Computing and Mathematics
Brunel University
Uxbridge, UK
e-mail: {bachar.alrouh, george.ghinea}@brunel.ac.uk

Abstract—Recently, web services security has shown a significant gesture as several specifications have been developed and implemented to meet the security challenges of web services. However, the performance of the security mechanisms is fraught with concerns due to additional security contents in SOAP messages, the higher number of message exchanges to establish trust as well as extra CPU time to process these additions. In this paper, we consider and compare the performance of various security mechanisms applied on a simple web service tested with different initial message sizes. The test results show that transport layer security mechanisms are considerably faster than message level security mechanisms. Moreover, the effect of adding SAML-tokens is negligible and the performance of SAML-based web services depends mostly on the underlying security mechanisms. Finally, the performance penalty of applying STS security mechanisms is significantly high comparing to non-STS mechanisms.

Keywords—Web Services; Security; Performance; WSIT

I. INTRODUCTION

Web services provide a platform neutral and programming language independent technology that supports interoperable machine-to-machine interaction over a network. It has an interface described using XML artifacts, such as the Web Services Description Languages (WSDL). Clients and other systems interact with the web service using a standardized XML messaging system, such as Simple Object Access Protocol (SOAP), typically conveyed using HTTP with an XML serialization in conjunction with other related web standards [1].

Nevertheless, the idea of applications from different parties communicating together raises a security threat. Message exchange security is an important issue to be taken into consideration in web services. The recipient of the message should be able to confirm its integrity and assure that it has not been modified. The message should be delivered to the recipient confidentially where only the authorized users could read it, know the identity of the sender and determine the operation requested in the message [2]. The challenge of web services security is to understand and consider the risks of securing a web-based service depending on the existing security techniques and simultaneously follow evolving standards in order to fill the gap in web services security [3].

As the interaction between service providers and requesters occurs via XML-based SOAP messages, securing web services tends to make these messages longer than they would be otherwise and consequently requiring interpretation by XML parsers on both sides, which reduces the performance of web services [4].

In this paper, we explore the effect of applying various WSIT security mechanisms on the performance of web services. The rest of this paper organized as follows: Section 2 discusses related works. Section 3 defines the Web Services Interoperability Technology. The test design is described in section 4. We present and analyze the test results in section 5. Conclusion remarks and future work are in section 6.

II. RELATED WORK

Performance is an open problem in web services, and has been analyzed in different manners. The majority of the studies have compared the performance of different toolkits. In [5] several SOAP toolkits have been evaluated with an objective of identifying and measuring SOAP inefficiency, while compared other toolkits by focusing on scientific data in order to raise the features of SOAP that affect web services performance. Another study [6] have compared the performance of web services with other middleware such as CORBA and Java RMI. This study has shown that the performance of web services is a major drawback, especially when applying security and reliability assertions.

In security related studies, Moralis et al [7] have compared the performance of web services with Kerberos Token Profile against X.509 Token Profile, while Liu et al [8] have conducted several tests for different operations (Signing vs. Verifying and Encryption vs. Decryption with several algorithms). The study of Tang et al [9] has compared the cost of WSS Signing and WSS Encryption. They also illustrated that using Username or X509 does not make significant difference in performance.

Shirasuna et al [10] have evaluated security mechanisms for grid services. Their evaluation has shown that transport level security is faster than message level security, and should be used if there is no additional requirement to use message level security.

Since WSIT [11] is relatively new, in our best knowledge, there is no published work that conducted performance evaluation of applying WSIT Security Mechanisms when writing this paper.
This paper focuses mainly on the overall performance of WSIT security mechanisms by studying the round trip time (RTT) from the client perspective. Therefore the following discussion does not essentially cover the performance of underlying WSS specifications implemented within the security mechanisms.

III. WEB SERVICES INTEROPERABILITY TECHNOLOGY

In order to improve web services’ Quality of Service (QoS) and to enable interoperability between Java and .Net web services, the Web Services Interoperability Technologies (WSIT) [11] has been developed as joint effort between Sun and Microsoft. WSIT is an implementation of a number of open web services specifications to support enterprise features, such as message optimization, reliable messaging, and security.

Web services have relied on transport-based security such as SSL to provide point-to-point security. WSIT implements WS-Security so as to provide interoperable message content integrity and confidentiality, even when messages pass through intermediary nodes before reaching their destination endpoint. WS-Security as provided by WSIT is an addition to the existing transport-level security, which may still be used.

While applying WSIT security mechanisms to enhance the security of web services, this may also result in increasing the size and number of the exchanged SOAP messages, which may in turn lead to an increase in the time of processing these messages and transmitting them over the network.

IV. TEST DESIGN

A. Test Scenario and Cases

The aim of this test is to study the effect of individual security mechanisms on web services performance. Therefore, we have designed and implemented a simple echo scenario to reduce the side effects of unrelated processing of the business logic. We use a simple JAX-WS echo application, which consists of a web service and a client. This scenario represents the peer-to-peer mode test; the client sends different size messages (from 1 Byte to 1MByte) and the web service echoes (send back) the same message received. The test has been run with and without applying security mechanisms, using different initial message sizes: 1byte to 1 Mbyte. Fig. 1 illustrates the security mechanisms tested and evaluated in this paper.

B. Test Environment and Settings

In this paper we focus on the increment of processing time when applying security mechanisms instead of the network latency. As a result, our data are collected from a local machine; the web service and the client are deployed on a Dell machine (Pentium D CPU 2.80 GHz / 3GB of RAM) Running Microsoft XP.

NetBeans IDE 6.5 is used to develop the web service and the client. The web service is developed as a web application and deployed on a Glass Fish 2.2 application server. We used a Java SE application to represent the client. The initial data sent from the client to the service are randomly generated before sending the message to avoid any caching. Metro’s WSIT web service stack 1.4 is used to apply security mechanisms to our web service.

C. Evaluation Metric

We measure, using Java’s System.nanoTime(), the time spent in requesting and responding on the client side as round trip time (RTT). We run every test 1000 times and calculate the average RTT for each case. The test then is repeated on 10 different occasions. The results shown in this paper represent the total average after eliminating the highest and lowest averages. We compare the results using the Round Trip Time Increment Percentage [9, 12] (RTTIP) in order to evaluate the performance overhead for a specific security mechanism deployment:

\[
RTTIP = \frac{RTT_i - RTT_0}{RTT_0} \times 100\% , \text{ where:}
\]

- RTT_0 is the round trip time without applying any security mechanism deployment.
- RTT_i is the round trip time of the web service with a specific security mechanism i deployment. In our test, we use the following deployments as test cases:
  - UA: Username Authentication with Symmetric Key.
  - UDP: Username with Digest Passwords.
  - MCS: Mutual Certificates Security.
  - SA: SAML Authorization over SSL.
  - SV: SAML Sender Vouches with Certificates.
  - STS: STS Issued Token.

Figure 1. Security Mechanisms
V. RESULTS AND ANALYSIS

In this section, we analyze the results using different criteria: security layer (transport vs. message), encryption type (symmetric vs. asymmetric), using SAML tokens and finally authentication type (direct vs. STS).

Table 1 lists all the measured results (in milliseconds) of the previous test cases.

<table>
<thead>
<tr>
<th>Security Mechanism</th>
<th>Initial Message Size (byte)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>No Security</td>
<td>24</td>
</tr>
<tr>
<td>UA</td>
<td>81</td>
</tr>
<tr>
<td>UDP</td>
<td>80</td>
</tr>
<tr>
<td>MCS</td>
<td>119</td>
</tr>
<tr>
<td>SSL</td>
<td>42</td>
</tr>
<tr>
<td>SA</td>
<td>51</td>
</tr>
<tr>
<td>SV</td>
<td>121</td>
</tr>
<tr>
<td>STS</td>
<td>266</td>
</tr>
</tbody>
</table>

A. Transport Security vs. Message Security

Fig. 2 illustrates the huge difference in performance between message level security, represented by UA, and transport level security using SSL. While the increment percentage of RTT using message level security increases when the data size increases, we can notice that it is decreasing when using transport layer security. The main justification is that SSL is lightweight because it does not involve any XML parsing. Therefore, transport layer security should be used if there is no special requirement to use message level security, such as having a chain of web services, and end-to-end security is required.

B. Username Tokens vs. Mutual Certificates

As shown in Fig. 3, when the initial data size is in the range of 1 byte to 1 Kbyte, the round trip time is increased by around 220-230% when applying username authentication mechanisms. The UDP performs slightly better than UA because the username token in UDP, unlike UA, is not encrypted due to the use of digest passwords. On the other hand, using MCS for the same data sizes increases the round trip time by 370-390%. The difference can be related to the fact that UA and UDP use symmetric key cryptography while MCS uses asymmetric cryptography, where symmetric is always faster than asymmetric encryption [11].

On the other hand, when large messages are exchanged between the client and the service (i.e. 1Mbyte) we notice that the difference between the performance of UA, UDP and MCS decreases dramatically because most of the processing time is spent on applying the actual encryption rather than manipulating the keys.

C. SAML: Over SSL vs. Mutual Certificates

Fig. 4 confirms that the performance of SAML-based security mechanisms (SA and SV) depends mainly on the underlying security method that is used to protect the data (SSL and MCS respectively).
In both of the security mechanisms, SA and SV, the sender vouches a SAML token for authorization. However, SA protects the exchanged message using SSL in the transport layer, while SV depends on mutual certificates, which is a message level security. Comparing the performances of (SSL vs. SA) and (MCS vs. SV) indicates that there is a negligible increase in RTTIP when SAML is applied, especially when very large messages are exchanged.

D. STS vs. Non-STS

In Fig. 5, we compare between the performances of STS-based and non-STS (direct client-service authentication) security mechanisms. The non-STS based mechanism used for the comparison is UA.

Since the security of the client is dependent upon the security mechanism selected for the STS itself, not the service, the STS itself is secured using a separate UA mechanism.

Results show that using third party STS tokens affect the performance of the web service significantly. When the initial message size is between 1 byte – 1 Kbyte, the RTTIP of STS is about 4 times its value when applying non-STS security mechanism, UA. Therefore, STS security mechanisms should only be used when the service and the client are in two different domains, where direct authentication can be an issue. However, when the initial data size reaches 1 M byte, the performance difference decreases noticeably.

VI. CONCLUSION

In this paper, we have compared the performance of several security mechanisms for web services. Our performance evaluation has shown that mechanisms that use transport level security are always faster than message level security mechanisms. In addition, Message level security protocols have a scalability problem if large messages are exchanged, unlike SSL-based mechanisms. Within message level security mechanisms, username authentication based mechanisms perform better than mutual certificates security. However, the difference is insignificant when using very large size messages. Using digest passwords instead of encrypting the whole username token can slightly improve the performance. The performance penalty of using SAML is very small and depends primarily on the underlying security mechanism. Finally, the performance of STS security mechanisms is massively less than non-STS mechanisms and should only be used when service and client are on different domains.

We plan to extend this work by testing the server side aspects of performance as well as including more security mechanisms and testing them in various conditions, such as different networks and security configurations.

REFERENCES