EVALUATION OF A SINGLE WS-SECURITY IMPLEMENTATION

M. Shopov, N. Kakanakov

Department Computer Systems, Technical University of Sofia, branch Plovdiv, Plovdiv 4000, 61 Sankt Petersburg Blvd., tel. +359 32 659 758, e-mail: mshopov@tu-plovdiv.bg, kakanak@tu-plovdiv.bg

Abstract: The paper presents an evaluation of a single Web Services Security implementation. The implementation is based on Apache Tomcat2 server and Axis2 toolkit. Evaluation is made for different security mechanism, including transport-level security (SSL/TLS), message-level security – XML encryption and XML digital signatures. Communication delay, encryption delays and message overhead are measured to analyze the performance in different scenarios. The experiments are taken out in local area network to exclude random influence from routing and transportation in Internet. Some of the experiments are taken in specially built environment (echo services) and some in real-working multi-tier system for distributed automation (getTemperature services) to provide data for comparison of the influence of the environment.

Key words: Web Services, WS Security, Secure communication, E-Business, E-Automation.

INTRODUCTION

Design and development of distributed systems goes in a new direction over the past few years – towards standardization, openness and integration with other business entities. High-level programming languages, component-based platforms, Internet technology, and standardized communication interfaces, all influence their development. Web services are one of the most promising technologies for building distributed systems that has the potential of becoming the core of a new Web-based middleware platform, providing interoperability between computational services. In this specific context security is very important feature. But security here has its specific requirements. The highly distributed, shared and dynamic nature of web services brings new requirements that the conventional web security mechanisms are not completely applicable to. The family of standards around WS-Security issued by OASIS [16] is targeting to solve the security problems with web services, but they are still too complicated and not enough trusted. Sample implementations of services using WS-Security (Web service security - WSS) mechanisms are shown in this paper. A performance analysis and a comparison with other security mechanism are also presented.

BACKGROUND

There are three fundamental security mechanisms that are also applicable to web services. These are transport level security, message level security, and role-based security [1, 5].

Transport level security includes SSL/TLS, basic authentication through username and password or combination of above. The advantages are that they are proven and trusted, supported by most clients and servers, understood by people responsible for their administration. Disadvantages are that the information is not protected after the transport end point and intermediate devices like firewalls do not have access to the content.

Message level security includes XML encryption [15], XML Signature [14], and security tokens. Advantages of this mechanism are that it allows messages to be self protecting (all the way to its consumer), parts of the message can be secured to different parties using separate keys for each. Disadvantages are that it is still not fully trusted and it uses other standards like XML Encryption and XML Signature that makes it more complex.

Role-based security is concerned with different privileges for different users. Every user is assigned a role and can access all resources associate with that role.

WS-Security describes enhancements to SOAP messaging to provide quality of protection through message integrity, message confidentiality, and single message authentication [3, 5, 16]. These mechanisms can be used to accommodate a wide variety of security models and encryption technologies. They can be combined in various ways to accommodate building a wide variety of security models using a variety of cryptographic technologies. All the security related information targeted to a specific recipient is included in the *<wss:security>* security header block that is attached within the SOAP header. More than one security header blocks can exist, if the message is targeted to more than one recipient.

WS-Security also provides a general-purpose mechanism for associating security tokens with messages. Additionally, WS-Security describes how to encode binary security tokens and attach them to SOAP messages. Specifically, the WS-Security profile specifications describes how to encode Username Tokens, X.509 Tokens, Security Assertion Markup Language (SAML) Tokens , and Kerberos Tokens as well as how to include opaque encrypted keys as a sample of different binary token types [2, 10].

Message integrity is provided by leveraging XML Signature and security tokens to ensure messages have originated from the appropriate sender and were not modified. Similarly, message confidentiality leverages XML Encryption and security tokens to keep portions of SOAP message confidential.

RELATED WORK

The authors of [4] have suggested comparison of performance of Web services and RMI (Remote method invocation), both secure and non-secure variants. The results show that with including of WSS the delay and relative message size increase significantly.

Both [7] and [8] are performing analysis of WSS. The time for encrypting/decrypting of messages with different cryptographic algorithms and the overhead introduced are measured. The experiments are made not only for different message sizes, but for different complexity of message structure. In [9] the comparison is made against two different security token profiles: Kerberos and X.509.

TESTBED ARCHITECTURE AND SCENARIOS

The test-bed architecture consists of a server, clients and an embedded controller with temperature sensor. The server is a Linux Debian 2.6.18-4-686 running on a machine with Intel(R) Pentium(R) 4 CPU 3.00GHz and 1GB of RAM. The JVM version used is jre1.6.0_01. Web services are running on Apache Tomcat 5.5.23 Web server with Axis2 [11] Web service engine and WSS4J [12] toolkit. The client software is running on machines with Linux and Windows XP. There two implementations – in Java and .NET. The embedded controller is IPC@Chip [13] with attached temperature sensor and Ethernet interface. The server, clients and controller are working in a 100Mbps Fast Ethernet LAN. For the packet capture and analysis a wireshark v.0.99.4 is used. The test-bed architecture is shown on figure 1.

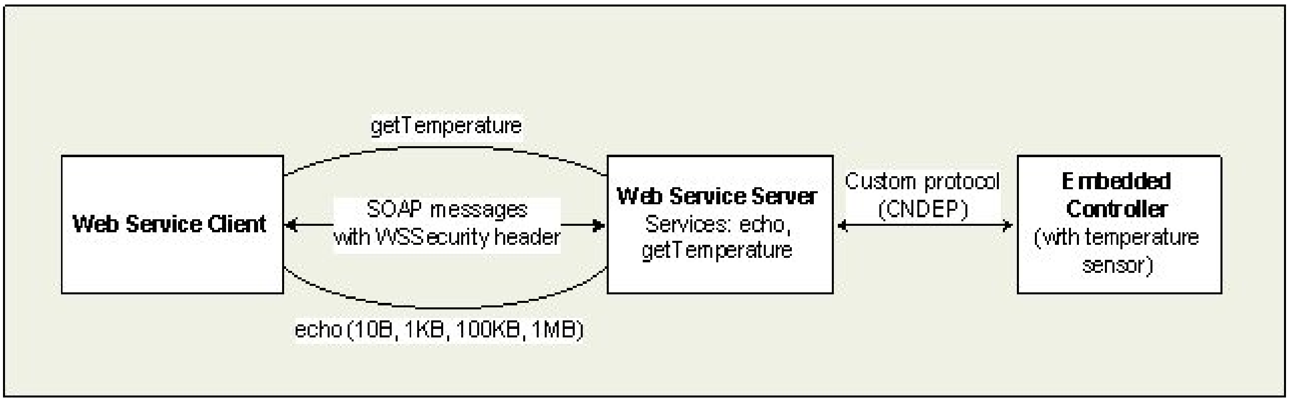

getTemperature
Web Service Client
SOAP messages with WSSecurity header
Web Service Server
Services: echo, getTemperature
Custom protocol (CNDEP)
Embedded Controller
(with temperature sensor)
echo (10B, 1KB, 100KB, 1MB)

Figure 1: Test-bed architecture

Four different scenarios were tested in the experiments. For each of them two sample services are used – echo and getTemperature. The echo service is a simple one. It accepts a string and reply with that string. It is tested with three different message sizes (1KB, 100KB, 1MB). The getTemperature service uses custom UDP based protocol – CNDEP [6] to obtain and return real-time temperature from the embedded controller (figure 2).

The first scenario *(scenario1)* uses only transport layer security. It is based on a TLS connection between the client and the Tomcat Web server hosting the web services. Next three scenarios use XML Encryption and XML Signatures. *Scenario2* uses XML encryption, encoding and decoding the messages with a shared key pairs. The X.509 secure tokens are used. *Scenario3* uses XML Digital signing to ensure the message was not manipulated. The last scenario (*scenario4*) uses a combination of both – first signing the message and then encrypts it.

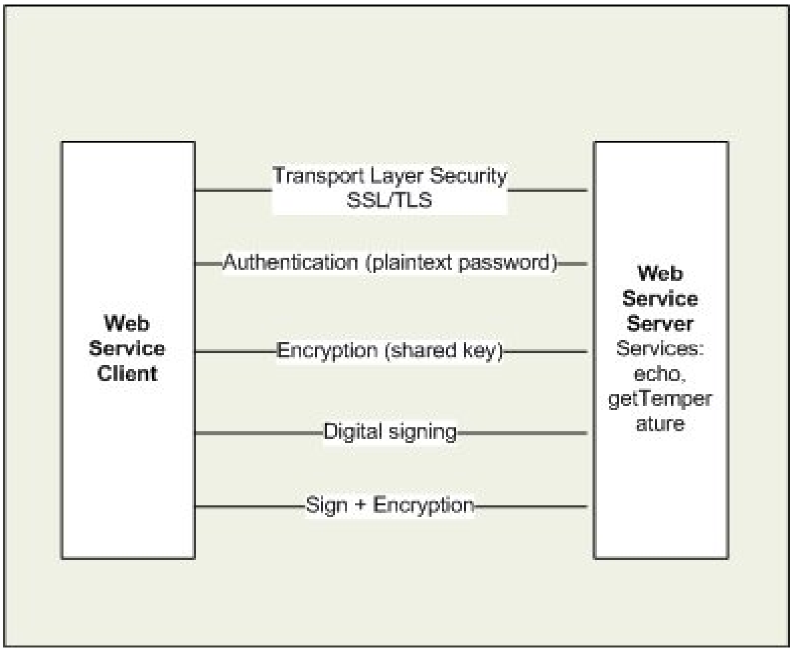

Web Service Client
Transport Layer Security SSL/TLS
Authentication (plaintext password)
Encryption (shared key)
Digital signing
Sign + Encryption
Web Service Server
Services: echo, getTemperature

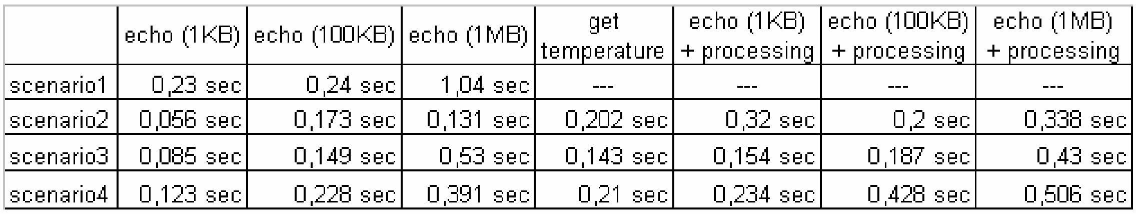
Figure 2: Experimental scenarios

EXPERIMENTAL RESULTS

This section presents some experimental results based on the experimental scenarios described in previous section. The aim of the experiments is to gather some time values for the processing of SOAP messages with security header and compare these times and message overhead with those for classical SSL/TLS case.

Next results show the induced from the WSS security header overhead in SOAP message. It is around 50% when only singing or encrypting the message and almost 80% when both are used (table 2, figure 4). In the experiments we have only signed and encrypted the whole message. Encrypting part of the messages for different web service's consumers will require additional security headers that are expected to increase the message overhead even more.

Table 1: Average processing time for four different scenarios and five different requests



	echo (1KB)	echo (100KB)	echo (1MB)	get temperature	echo (1KB) + processing	echo (100KB) + processing	echo (1MB) + processing
scenario1	0,23 sec	0,24 sec	1,04 sec	---	---	---	---
scenario2	0,056 sec	0,173 sec	0,131 sec	0,202 sec	0,32 sec	0,2 sec	0,338 sec
scenario3	0,085 sec	0,149 sec	0,53 sec	0,143 sec	0,154 sec	0,187 sec	0,43 sec
scenario4	0,123 sec	0,228 sec	0,391 sec	0,21 sec	0,234 sec	0,428 sec	0,506 sec

The results present the average values collected after 101 executions. Table 1 summarises the average processing time for all scenarios and services. Graphical representation is given on figure 3. Experiments with TLS (scenario1) are made only with the echo service.

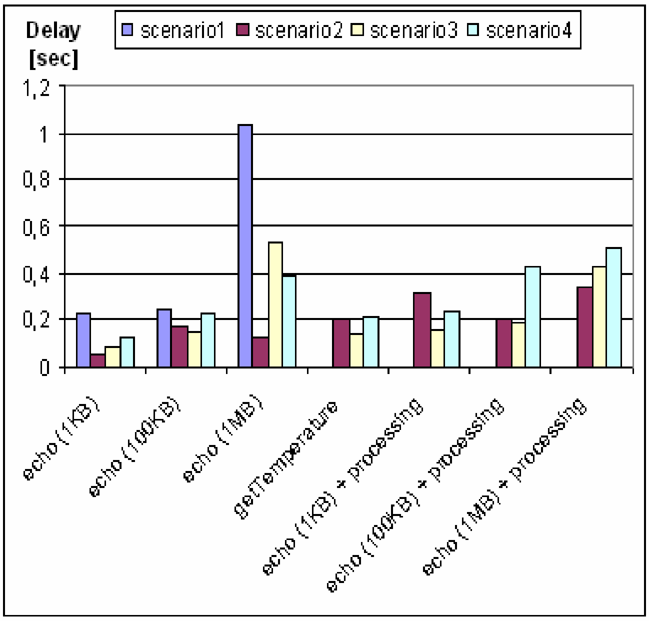
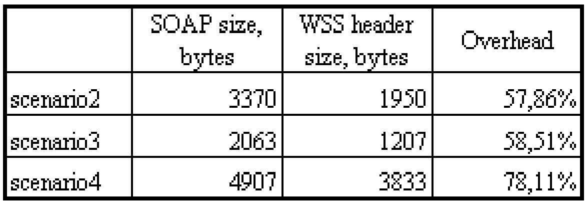


Figure 3: Average processing time for four different scenarios and five different requests

As observed on figure 3, the transport layer security has worst performance compared to the scenarios in which WS-Security is used. It is in order of magnitude slower for the longer message sizes – 1MB.

Table 2: WSS overhead to SOAP messages



	SOAP size, bytes	WSS header size, bytes	Overhead
scenario2	3370	1950	57,86%
scenario3	2063	1207	58,51%
scenario4	4907	3833	78,11%

The result shows also that the processing time of XML signature is not fully dependent of the message size like it is for the XML message encryption (figure 3).

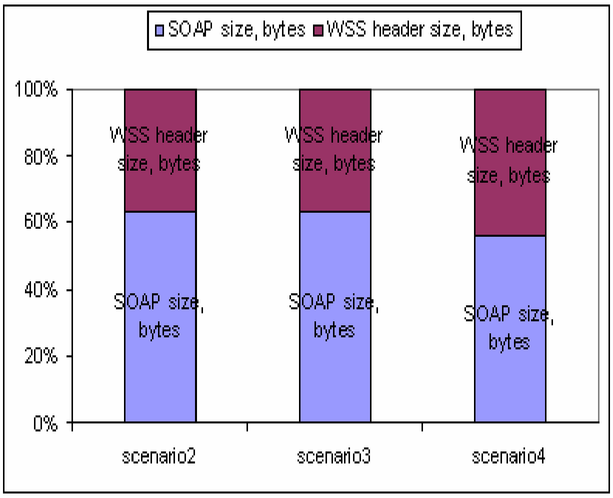


Figure 4: Relative message overhead introduced by WSS security header

CONCLUSIONS

Web Services Security (WSS) is very specific kind of security and cannot be evaluated as traditional application security. The communications security, the application security and message security are integrated in a single specification for meeting the complex needs of e-world security. Most of the methods used in WSS are based on traditional ones used in securing information. For general estimation of the WSS performance, the different aspects must be estimated separately to prevent interference of the experimental results. In this paper, a practical performance evaluation of the message and communication security provided by WSS is made. The measured parameters provide base for evaluation the communication overhead and delay, which is very important for time-critical services. Transport level security provides only an end-to-end secure communicational channel, which is applicable only for simple services. For more complex services that interact in complex program environment, the security must be provided directly in the messages. This provides protection of the message contents, not only during transport but in the application itself. The message or only parts of it can be encrypted, digitally signed or both. This scalability helps the security administrator to make a balance between performance and security. The main conclusion is that message-level security provides more scalable way for protection than transport-level security and even work faster.

The future work will be directed in evaluation of the other parts of the complex estimation of the security. The main work must be in estimation of context-based filtering and availability of services.

ACKNOWLEDGEMENTS

The presented work is supported by National Science Fund of Bulgaria project – “MU-MI-1602/2006”, under the contract “D01-777/26.10.06”.

REFERENCES

[1] Ganesan, H., “Web Services: Interoperability and Security,” Arizona State University, Students Report, 2004.

[2] Geer, D., "Taking Steps to Secure Web Services," Computer, vol.36, no. 10, pp. 14-16, Oct., 2003.

[3] Gutierrez, C., E. Fernandez-Medina, M. Piattini, „Web Services Security: Is the problem solved?,” INFORMATION SYSTEMS SECURITY, vol.13; part 3, pp. 22-31, 2004, ISSN 1065-898X.

[4] Juric M., I. Rozman, B. Brumen, M. Colnaric, M. Hericko, “Comparison of performance of Web services, WS-Security, RMI, and RMI–SSL,” Journal of Systems and Software, vol. 79, pp.689–700, 2006.

[5] Hratmann, B., J. Flinn, B. Konstantin,S. Kawamoto, “Mastering Web Services Security,” Wiley Technology, Indianopolis, 2003, ISBN: 0471267163.

[6] Kakanakov, N., I. Stankov, M. Shopov, G. Spasov, Controller Network Data Extracting Protocol – design and implementation, Proc. Computer Systems and Technologies Conference, 2006, pp.IIIA-14-1 – IIIA-14-6.

[7] Kezhe Tang, Shiping Chen, David Levy, John Zic, Bo Yan, "A Performance Evaluation of Web Services Security," edoc , pp. 67-74, 2006.

[8] Liu, H., Pallickara, S., and Fox, G., Performance of Web Services Security, in Proceedings of 13th Annual Mardi Gras Conference, Feb. 2005.

[9] Moralis A., V. Pouli, M. Grammatikou, S. Papavassiliou, V. Maglaris, “Performance Comparison of Web Services Security: Kerberos Token Profile Against X.509 Token Profile,” IEEE 3rd International Conference on Networking and Services, Athens Greece, June 19-25, 2007.

[10] Naedele M., “Standards for XML and Web Services Security,” Computer, vol. 36, no. 4, pp.96-98, Apr., 2003.

[11] Apache Axis - http://ws.apache.org/axis/

[12] Apache WSS4J - http://ws.apache.org/wss4j/

[13] Beck IPC GmbH, IPC@CHIP, Website: http://www.beck-ipc.com/ipc/, [June 2007].

[14] XML Signature Syntax and Processing: http://www.w3.org/TR/xmldsig-core/

[15] XML Encryption Workgroup: http://www.w3.org/Encryption/2001/

[16] http://docs.oasis-open.org/wss/v1.1/wss-v1.1-spec-os-SOAPMessageSecurity.pdf