PROCEEDINGS OF THE WORKSHOP ON APPLICATIONS OF SOFTWARE AGENTS
ISBN 978-86-7031-188-6, pp. 45 - 51, 2011

Security Based Performance Issues in Agent-based Web Services Integrating Legacy Information Systems

Sashko Ristov[1], Aristotel Tentov[2],

[1] Ss. Cyril and Methodius University / Faculty of Natural Science and Mathematics - Institute of Informatics, Gazi Baba BB,
1000 Skopje, Macedonia
sasko@ii.edu.mk
[2] Ss. Cyril and Methodius University / Faculty of Electrical Engineering and Information Technologies, Rugjer Boshkovik bb, PO Box 574,
1000 Skopje, Macedonia
toto@feit.ukim.edu.mk

Abstract. Many closed internal information systems (IS) must not concern much about its security. But, nowadays, mostly due to globalization and dynamic world, business demands integration of several legacy ISs into new one. The new IS must improve its security compared to legacy ISs, because they work in the heterogeneous environment, and need to be opened. Adding security will decrease the new IS performance, but business wants to retain or even improve it. In this paper, we simulate this issue and we create a baseline of performance data that can be used to predict IS response time, for various numbers of requests, request size and implementing security as a necessary issue for new distributed IS.

Keywords: Software agents, wrapper, web services, legacy software, performance, security

1 Introduction

Nowadays, there are many cases where companies' ISs, often developed using different technologies and on different platforms, need to communicate each other. Even more, many companies buy or merge with others, which lead in merging two independently developed ISs, probably with totally different structure. As a solution concept, [15] offers three basic approaches, shown on Fig. 1. The first one, introducing the *transducer*, accepts messages from the requester, translates them into the program's native language and protocol, and passes to legacy IS. The second one, introducing the *wrapper*, can "inject" code into legacy IS to allow it to communicate with the requester. The third solution that is, *rewriting* is out of scope of this paper. Many papers propose agent based web services as a solution for this issue, as a platform independent and XML as most obvious format that meets this requirement.

In most of cases, legacies ISs are often isolated and closed to outside environment, and do not concern much about security. However, the new IS must improve better

security compared to legacy ISs, because legacy ISs are often on a different locations and new IS will work in the heterogeneous environment, wide-opened to hackers.

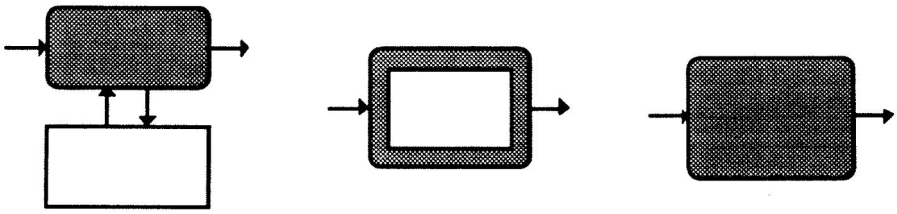

Fig. 1. Three approaches to agentification, Transducer, Wrapper, Rewrite

In order to secure the new ISs, has to be achieved message confidentiality, data integrity, authentication, authorization, non-repudiation, service availability, and web service identification. For web services, XML Encryption [10] and XML Signature [9] are considered as a de facto standard to provide a message security model. Their implementation secures the new IS, but outcomes with the message overhead, as well as requires complex cryptographic operations for each message or some parts of it. Thus, it reduces the new IS performance. It is expected that the new IS will have more users than legacy, which also impacts negative to its performance.

In this paper, we analyze the response time overhead of the signed and encrypted messages through experimental approach, as well as the response time overhead of increasing message size for the same message type. In Section 2, we show the results of the experiments and analyze the dependency of the new IS response time for a various number of requests. Next, in Section 3, we analyze the results of the experiments in order to gain the response time ratio implementing signature and encryption compared to legacy IS messages, and implementing encryption compared to signed only messages.

1.1 Related Work

We find many articles discussing about agent-based web services. [1] presents an agent-based web services evolution approach, which is well suited to building software solutions for pervasive computing. [3] describes how web services will become more agent-like and how the resultant agent-based web services will yield unprecedented levels of software robustness. In [13] is presented an integration of mobile agent and the web services technology and new security architecture for such integration. In [14] the authors propose an agent-based Web services architecture and apply to augment manufacturability to increase the efficiency of the distributed collaboration. [4] presents some agent-based and context-oriented approach for Web services composition. A gateway architecture for connecting software agents and Web services in a transparent manner with fully automatic operation is shown in [5]. [6] describes a possible solution for pro-active Web services selection and composition.

In [7] is introduced an architecture which seamlessly connects mobile agents with web services in a transparent and fully automated manner by means of a specialized

Web Service Engine. [8] shows how agent technology can be used to personalize web services and to work together.

One of the most near article is [11], where authors define major features and benefits of the agent-based approach to enhance a Legacy Information System. Also, [12] discuss how to re-engineer legacy systems into agent-based Web services and focuses on the fact that agent-based Web services are well suited to building software solutions for distributed, open and dynamic web-based systems.

Only in [2], authors compare two transducer-based approaches to wrapping legacy software, introducing that web services make less overhead of the messages comparing software agents.

None of the related work analyzed the performance impact of the agent-based web services, nor implementing different security-level mechanisms, using different message size and concurrent requests.

This paper describes a series of experiments focused on understanding the performance impact of message overhead, as well as increased number of requests, in simulated wrapped legacy IS. We create a baseline of performance data that can be used to predict new IS response time. That is, the number of concurrent requests can be predicted, as well as the message sizes, because the new IS system information has equally size to the legacy and the number of requesters can be predicted as a sum of all users in the legacy systems.

1.2 Testing Environment

We create three test environments, (1) using unsecured messages, (2) signed messages and (3) both signed and encrypted messages, on Windows platform. We simulate a situation where two parameters (let's say *Street* and *City*) in legacy IS are only one parameter (let's say *Address*) in the new IS. Thus, the wrapper receives and returns equally sized messages. In each test case, we change the message size and the number of concurrent requests, but in the range of normal workload.

2 Results and Analysis for Increasing Number of Requests

This section describes the results of the performed tests to measure the response time dependency of the message size and a given number of requests, for the same message type. On Fig. 2 is shown the response time in milliseconds (Y axis), for a given number of requests in second and message type, for a different message size in kilobytes (X axis).

2.1 Response Time Overhead for no Security

First, we perform a performance baseline, that is, we analyze the response time without security for different payload of 1, 10 and 100 messages per second for different message size. As shown on Fig. 2, the results are very strange. Namely, for small sized messages, the system has better performance when loaded with 10 or 100

messages per second, and for bigger messages (over 30K), the system performance is as expected, that is, higher response time for a huge payload and bigger messages.

2.2 Response Time Overhead for Signed Messages

Next, when we implemented signature into messages, we analyze the response time for different payload of 1, 10 and 100 messages per second. As shown on Fig. 2, the results are a little strange. Namely, for small sized messages (smaller than 35K), the system has better performance when loaded with 10, instead of loaded with 1. For a 100 messages per second, the system performance is as expected, that is, higher response time for a huge payload and bigger messages.

2.3 Response Time Overhead for Signed and Encrypted Messages

Next, we implemented the encryption, besides signature, and payload the system with 1, 10 and 100 messages per second. As shown on Fig. 2, the results are similar as signed only messages. Namely, for small sized messages (smaller than 25K), the system has better performance when loaded with 10, instead of loaded with 1. For a 100 messages per second, the system performance is as expected.

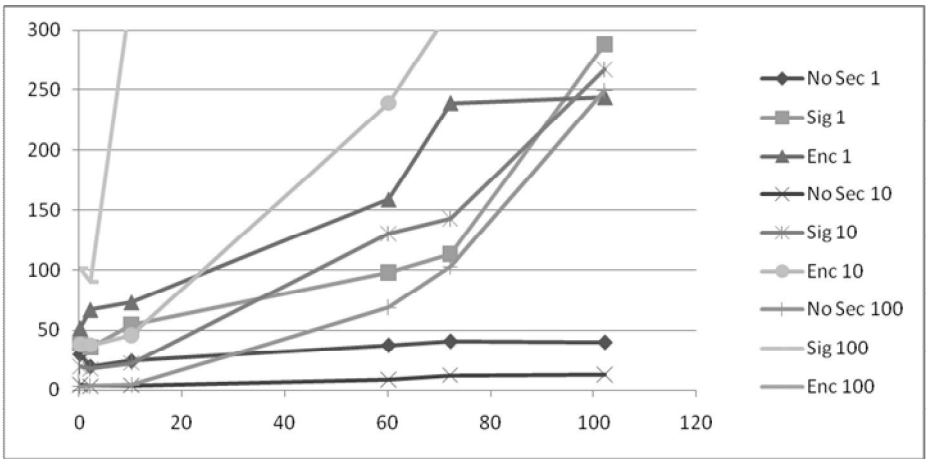

Fig. 2. Response time in ms for a given number of requests in a second and message type, depending of message size

3 Results and Analysis for Different Message Security Type

This section describes the results of the performed tests to measure the response time overhead of the security implementation, for a given number of requests and various message size. On Fig. 3 is shown the response time ratio (Y axis) of implementing

signature and both signature and encryption to the unsigned messages, for a given number of requests, and different message size in kilobytes (X axis).

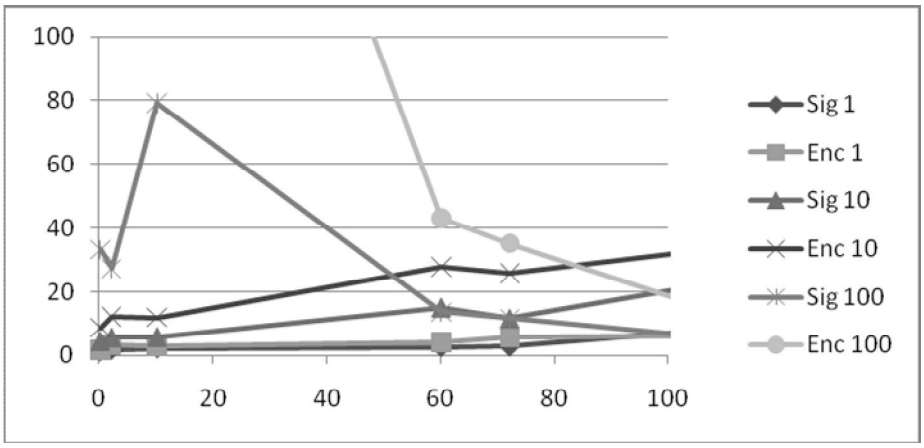

Fig. 3. Response time overhead (ratio) for implementing security for a given number of requests, depending of message size

3.1 Response Time Overhead Implementing Security

We analyze the response time overhead implementing signature to the different sized messages, compared to the same unsigned message for different payload of 1, 10 and 100 messages per second. As shown on Fig. 3, adding signature increases the response time ratio constantly, near linear, growing slowly when increasing message size, but only for a small number of requests, because for a huge number (100), the baseline payload (without security) has huge response time (shown on Fig. 2).

Implementing both signature and encryption creates similar overhead to the no security messages.

3.2 Response Time Overhead Adding Encryption

At the end, we analyze the response time overhead adding encryption to the signature for different sized messages, compared to the same signed only message for different payload of 1, 10 and 100 messages per second. On Y axis on Fig. 4 is shown the response time ratio, and X axis is original message size.

We can conclude that adding encryption to the signature increases the response time ratio constantly, near linear, but only for a message size above 10K, growing slowly when increasing message size. The reason that the ratio decreases at the range of a huge messages and especially huge number of requests is because in that range, the baseline, that is the signed messages only response time, is huge. With other words, the IS has a low performance for that payload.

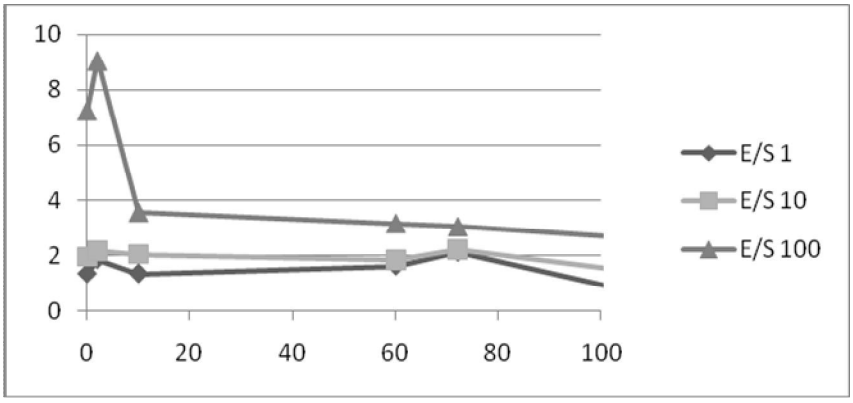

Fig. 4. Response time overhead (ratio) for adding encryption to the signature

4 Conclusion

In this paper we have done security based performance analysis and comparison simulating a small part of new IS, which operates as a wrapper to the legacy IS. We analyze the performance parameter "response time", as well as its overhead increasing "message size", "number of messages", and implementing different security-level mechanisms.

We believe that these result can be considered as a baseline and with these analysis, IT quality managers can predict response time for a new IS, knowing number and size of concurrent messages, in order to retain the new system secured, as the legacy.

5 Future Work

In this paper we analyzed the decreasing the performance to the new IS, implementing security, increasing "message size", "number of messages", and implementing different security-level mechanisms. But, for the business managers, the most important issue is customer satisfaction, which is the most connected to the performance issue, or response time.

Therefore, our future work will be oriented on necessary hardware improvements to the new IS, to retain the same performance, such as the legacy system, for the same payload, but with implemented signature and encryption. Also, the operating system platform must be analyzed.

6 References

1. Liu, R., Chen, F., Yang, H., Chu, W.C., Lai, Y.: Agent-Based Web Services Evolution for Pervasive Computing. In: APSEC '04 Proceedings of the 11th Asia-Pacific Software Engineering Conference, pp 726--731. IEEE Computer Society, Washington, DC, USA (2004)
2. Oglodek, M., Gawinecki, M., Paprzycki, M.: Utilization of Software Agents and Web Services as Transducers for Legacy Software; Case Study Based on an SMTP Server. In: Proceedings of the International Multiconference on Computer Science and Information Technology, Vol. 2, Polish Information Processing Society, Wisla, Poland (2007)
3. Huhns, M.N.: Software Agents: The Future of Web Services. In: Kowalczyk, R. et al. (Eds.): Agent Technology Workshops 2002, LNAI 2592, pp. 1–18, 2003. Springer, Heidelberg (2003)
4. Maamar, Z., Mostefaoui, S.K., Yahyaoui, H.: Toward an Agent-Based and Context-Oriented Approach for Web Services Composition. J. IEEE Transactions on Knowledge and Data Engineering, Vol. 17, No. 5, 686--697, (2005)
5. Greenwood, D., Calisti, M.: Engineering Web Service – Agent Integration. In: IEEE Systems, Cybernetics and Man Conference, vol.2, pp 1918--1925, The Hague, Netherlands, (2004)
6. Matskin, M., Küngas, P., Rao, J., Sampson, J., Petersen, S.A.: Enabling Web Services Composition with Softward Agents. In IMSA (2005) pp 93--98
7. Peters, J.: Integration of Mobile Agents and Web Services. In: Proceedings of the first European Young Researchers Workshop on Service Oriented Computing (YR-SOC), UK, (2005)
8. Kuno, H., Sahai, A.: My Agent Wants to Talk to Your Service: Personalizing Web Services through Agents. In: HPL- 2002-114, HP Labs Technical Report, 2002.
9. Eastlake, D., Reagle, Solo, J. D., eds: XML-Signature Syntax and Processing W3C Recommendation, (2002), http://www.w3.org/TR/xmldsig-core/
10. Eastlake, D., Reagle, J., eds.: XML Encryption Syntax and Processing. W3C Recommendation, (2002), Online at http://www.w3.org/TR/xmlenc-core/
11. Nguyen, M. T., Fuhrer, P., Pasquier, J.: Enhancing Legacy Information Systems with Agent Technology. In: J. International Journal of Telemedicine and Applications - Special issue on electronic health archive, Vol. 2009, (2009)
12. Chen, F., Yang, H., Guo H., Xu, B.: Agentification of Web Services. In: COMPSAC '04 Proceedings of the 28th Annual International Computer Software and Applications Conference - Vol. 01, pp 514--519. IEEE Computer Society Washington, DC, USA (2004)
13. Zhang, J., Wang, Y., Varadharajan, V.: Mobile Agent and Web Service Integration Security Architecture. In: SOCA '07 Proceedings of the IEEE International Conference on Service-Oriented Computing and Applications. IEEE Computer Society Washington, DC, USA (2007)
14. Huang, C., Tseng, T., Gung, R. R., Chang H.: An agent-based web services solution to collaborative product design. International Journal of Knowledge-based and Intelligent Engineering Systems, Vol. 9, n. 2, p.63--79, (2005)
15. Genesereth, M. R., Ketchpel, S. P.: Software Agents. In: Communication of the ACM, Vol. 37, No. 7 July (1994)