Toward Systematic Integration of Security Policies into Web Services

Azzam Mourad
Department of Computer Science and Mathematics,
Lebanese American University,
Beirut, Lebanon
Email:azzam.mourad@lau.edu.lb

Hadi Otrok
Department of Computer Engineering,
Khalifa University of Science,
Technology & Research,
Abu Dhabi, UAE

Sara Ayoubi
Department of Computer Science and Mathematics,
Lebanese American University,
Beirut, Lebanon

***Abstract*—In this paper, we introduce our approach for the automatic generation of BPEL (Business Process Execution Language) aspects from security policies. It is based on a synergy between policies, Aspect-Oriented Programming (AOP) and composition of web services. Our proposed approach allows first to transform security policies into BPEL aspects. Then, the generated aspects are weaved in the BPEL process of the composed web services at runtime [1]. The main contributions of our approach are: (1) Describing dynamic security policies, (2) generating automatically the BPEL aspects, (3) separating the business and security concerns of composite web services, and hence developing them separately (4) allowing the modification of the dynamic security features and web services composition at run time and (5) providing modularity for modeling cross-cutting concerns between web services.**

Keywords. Web Services Security, BPEL, Security Policies, AOP, RBAC

I. INTRODUCTION

Web services made business applications more accessible over the internet. They do not just widen the broad of applications accessibility but it is also a catalysis for collaboration between several distributed applications through the composition concept. Nevertheless, the successful deployment of this technology cannot hide the security breaches and threats [2] that a web service can be exposed to. Therefore, enforcement of web service security is one of the most important duties, which the research community has to perform.

In this context, several standard languages have been proposed to enforce web services security. The Security Assertion Markup Language (SAML) [3], WS-Security [4] and WS-XACML [5] are the most successful ones. The main problem with language-based strategies is that the main focus is on embedding the security features in the design/code of the web services, i.e, integrating them statically. However, many security features require run-time verification of the security policies, which may often be modified and updated. This means that when the security policies and/or the verification strategy change, the developer has to go back to the design/code of the web services and update them accordingly.

This problem is raised more when several web services are composed together in a BPEL (Business Process Execution Language) process to form a more complex system. With the use of the current BPEL, there is a lack of modularity for modeling cross-cutting concerns and inadequate support for changing the composition at runtime. Any changes in the environment and addition or removal of partner web services requires static and dynamic adaptation. In other words, if a BPEL runtime process change is required, we have to stop the running process, change the needed web service(s), modify the composition, and then restart. Such a mechanism is cumbersome, error-prone and tedious.

We introduced in [1] an aspect-oriented approach for the dynamic enforcement of web services security. This requires the specification of security requirements directly as BPEL aspects. However, this is considered as a shortcoming due to the complexity of security policies and their composition of one or more combinations of rules. Developers face several problems while building the security policy, roles, procedures and permissions directly into the aspects and code. On the other hand, few approaches [3]–[5] explored the usefulness and efficiency of specifying security policies in standard languages. Moreover, using such languages enforce the concept of separation of concerns, which is one of the main advantage of our previous work.

This work is supported by the Lebanese American University and Khalifa University of Science, Technology & Research

In this paper, we propose an extension to our approach by generating systematically the BPEL aspects from security policies. Our proposition is based on a synergy between policies, Aspect-Oriented Programming (AOP) and composition of web services. AOP allows to specify the security concerns in separate components called aspects. These aspects are then weaved (integrated) in the BPEL process at runtime. The main contributions of our approach are:

1) Describing the security policies.
2) Generating automatically the BPEL aspects of the security policies.
3) Separating the business and security concerns of composite web services, and hence developing them separately.
4) Allowing the modification of the web services composition at run time to integrate, remove and/or update security mechanisms.
5) Providing modularity for modeling security cross-cutting concerns between web services.

The rest of the paper is organized as follows. In Section II, we discuss the related work. Section III is dedicated to the illustration of the proposed approach. In Section IV, we illustrate the implementation of our propositions in a case study. Finally, we give some concluding remarks in Section V.

II. Related Work

Web service security is one of the topics that attracted the attention of the research community. From the definition of standards to the publication of research papers, the goal is to provide policies and mechanisms for enforcing web services security. In this context, we first summarize and explore the advantages and limitations of the current standards for security policy description. Then, we present the current initiatives related to the adoption of AOP in the web services environment and discuss their objectives, advantages and shortcomings.

SAML [3] is a specification language that is proposed by OASIS. Based on XML, it defines how to specify security credentials, which are represented as assertions. WS-Security [4] is a standard that is proposed by IBM, Microsoft, and Verisign. WS-Security is a means for using XML to encrypt and digitally sign SOAP messages. The web Service eXtensible Access Control Markup Language (WS-XACML) [5] is proposed by OASIS as XML-based language to specify and exchange access control policies. Bhatti et al. [6] proposed X-RBAC: an XML-based RBAC policy specification framework for enforcing access control in dynamic XML-based web services. Damiani et al. [7] proposed a design of a web service architecture for enforcing access control policies. They also provide an example of implementation based on the WS-Policy [8], [9] as access control language. Bertino et al. [10] proposed a RBAC-WS-BPEL framework for defining authorization policies and constraints for WS-BPEL business processes.

The aforementioned related work support our claim about the need to have standard languages for the description and specification of security policies. However, the main problem with the proposed solutions for the enforcement of web services security is the static embedding of the security features in the design/code of the web services. In fact, many security features require run-time verification of the security policies, which may often be modified and updated. This means that when the security policies and/or the verification strategy change, the developer has to go back to the design/code of the web services and update them accordingly. Such a mechanism is cumbersome, error-prone and tedious. Our approach relies on the dynamic injection of AOP aspects into BEPL processes. This allows to easily update the security measures when needed, without affecting the business logic of the BPEL process.

In the sequel, we present the initiatives related to the adoption of AOP in the web services context. Anis Charfi et al. [11] introduced the AO4BPEL, an aspect oriented extension for BPEL that offers modularity and adaptability to workflow languages.

Mingjie Sun et al. [12] proposed an approach to monitor the WS-BPEL runtime information in order to improve its quality of service such as temporal logic (sequence of messages), timeliness (keeping track of the execution time of each activity) and reliability. Their approach uses the AOP standard to monitor the quality of service of the BPEL process, while our approach address security and allow the dynamic modification of the BPEL process.

Yunzhou Wu and Prashant Doshi [13] came with a new approach that aims to elevate the challenge of coordination between concurrent activities produced after the occurrence of several constraints. Mathieu Braem et al. [14] proposed an approach of advanced "Stateful aspects" in order to denote the crosscutting concerns that depend on the past states of the program. Both these approach offers extensibility to WS-BPEL and the ability to track the state of the process during its execution. However, our approach address security and allow the dynamic modification of the BPEL process.

III. Approach Description

Our proposed approach is based on a synergy between policies, Aspect-Oriented Programming (AOP) and composition of web services. Policy standard languages [3], [4] are very useful for the organized description of complex and composed security policies. They allow to avoid the ad-hoc description of security rules and specify them in XML-based document(s). On the other side, AOP [15]–[17] is one of the most prominent paradigms that have been devised for integrating non-functional requirements (e.g. security) into software. The main objective of AOP is to have a separation between cross-cutting concerns. This is achieved through the definition of aspects. Each aspect is a separate module in which pointcuts are defined. A pointcut identifies one or more join points. A join point identifies one or many flow points in a program (in our case a program is a BPEL process). At these points, some advices will be executed. An advice contains some code that can alter the process behavior at a certain flow point. The integration of aspects within the application code is called weaving and is performed through one of the weaving technologies (e.g., AspectJ [17]).

Security is one of the software aspects that are very important to deal with. Generally, developers describe in ad-hoc manner the required security rules and integrate them directly in their code. Moreover, they do not separate between security and business logic code. This means that any change in the security strategy has to be done on the application code, which can have impact on the business logic of web services. Security policies and AOP contribute to solve these issues by specifying the security policies in XML-based documents, then embedding them in aspects. Aspects allow to precisely and selectively define and integrate security objects, methods and events within application, which make them interesting solutions for many security issues. Many contributions [18], [19], in addition to our experiments [20], have proven the usefulness of AOP for integrating security features into software. Moreover, previous approaches [3]–[5] explored the usefulness and efficiency of specifying security policies in standard languages. Using such languages also enforce the concept of separation of concerns, which is one of the main advantages of AOP.

In this context, we present in this section an approach for the systematic transformation of security policies into BPEL aspects. These aspects are expressed using our elaborated language AspectBPEL (introduced in [1]) and then weaved in the BPEL process at runtime. The proposed approach is depicted in Figure 1. It illustrates the BPEL process where the invoke of the security features is embedded in BPEL aspects generated automatically from the security policies. In order to weave the generated BPEL aspects into a specified BPEL process, we have elaborated a language called AspectBPEL and developed its corresponding framework and weaver.

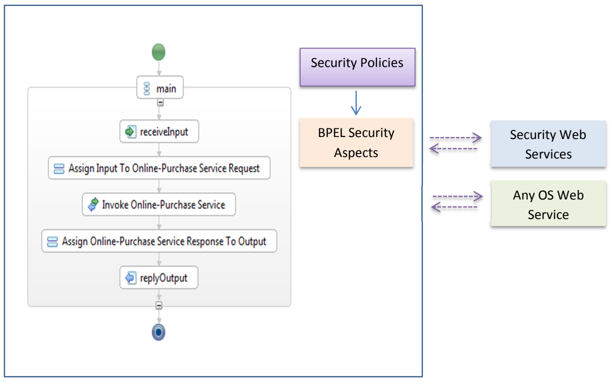
main
receiveInput
Assign Input To Online-Purchase Service Request
Invoke Online-Purchase Service
Assign Online-Purchase Service Response To Output
replyOutput
Security Policies
BPEL Security Aspects
Security Web Services
Any OS Web Service

Figure 1. Approach Schema

We can see in Figure 1 that the security features are no longer part of the web services, and hence any modification in the security policies can be reflected in the generated BPEL aspects that are weaved dynamically in the system BPEL process. The BPEL aspects may contain direct security verification to be integrated at some identified join points in the BPEL process, or it may contain an invoke to external web service(s) that handle the security verification.

IV. Generated User Authentication BPEL Aspect

In what follows, we describe a generated BPEL aspect for authentication. Please note that the syntax and constructs of the aspects are specified in AspectBPEL language. Our ApsectBPEL framework will compile the aspects. Our approach has been tested and we were successfully able to integrate security features into a BPEL process. Listing 1 illustrates excerpt of an aspect to authenticate the user and assign him role and permission(s). We define the location identifier of the Authentication behavioral code to be after the <bpel:receive...> construct that receives the request from the user. The Authentication aspect code consists first of assigning the client login info to the web service request message. Then, the web service is invoked. This latter calls the UserAuthentication operation that loops through the database and returns one of four possible indexes: 1 if the client's username is invalid, 2 if the client has entered correct username and password, 3 if he entered valid username but incorrect password or 4 if he has no more trials and is not allowed to login anymore.

Listing 1. Aspect for Authentication

```
Aspect Authentication

BeginAspect

After  <bpel:receive name="receiveInput" partnerLink="
     client" .../>
BeginBehavior
<!-Initialize Authentication Request Message-->
<bpel:assign validate="no" name="Assign Input to
     Authentication message">
   ...

<!-Invoking the Authentication WS-->
<bpel:invoke name="Invoke Authentication" partnerLink
     ="Authentication" operation="userAuthentication"
     portType="ns:Authentication" inputVariable="
     AuthenticationRequest" outputVariable="
     AuthenticationResponse">
</bpel:invoke>

<!-- if User is InValid, Reply with error message and
     Exit Process-->
<bpel:if name="If Invalid User">      <bpel:condition
     ><![CDATA[($AuthenticationResponse.parameters/ns:
     userAuthenticationReturn!=2)]]>
</bpel:condition>
<bpel:sequence>

<!-Initialize the ErrorString message-->
...
<!-Copy the AuthenticationResponse to the ErrorStr -->
...
<!-Invoke GetErrorStr WS-->
...
<!-Initialize Bpel Output Message-->
...
<!-Copy Authentication ErrorStr to Output variable-->
...
<!-Return "User Not Authorized" to the Client"-->
<bpel:reply name=" Return ErrorString" partnerLink="
     client" operation="process" portType="tns:
     AnyWSProcess" variable="output">
</bpel:reply>
</bpel:sequence>
</bpel:if>
EndBehavior
EndAspect
```

V. Conclusion

A new approach was introduced for the systematic transformation security policies into BPEL aspects for the dynamic enforcement of web services security. Our proposition is based on a synergy between policies, AOP and composition of web services. It allows policy specification and separation between business and security concerns of composite web services, and hence developing them separately. It also allows the modification of the web services composition at runtime and provides modularity for modeling cross-cutting concerns between web services. The experiments resulting from generating the policies corresponding BPEL aspects, and then deploying them dynamically in the BPEL process of the online purchase system, demonstrate the feasibility and appropriateness of our propositions. They also illustrate the successful dynamic integration and modification of authentication features in the online purchase system.

References

[1] A. Mourad, S. Ayoubi, H. Yahyaoui, and H. Otrok, "New Approach for the Dynamic Enforcement of Web Services Security," in *Proceedings of the 8th Conference on Privacy, Security and Trust(PST)*, Ottawa, Ontario, Canada, August 2010.

[2] K. N. Computing, "XML and Web Services: Message Processing Vulnerabilities," http://www.webservicessummit.com/Articles/MessagingThreats.htm.

[3] B. Lockhart and al., "OASIS Security Services TC (SAML)," http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security.

[4] B. Atkinson and al., "Web services security (WS-Security)," http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss.

[5] T. Moses, "OASIS eXtensible Access Control Markup Language (XACML), OASIS Standard 2.0," http://www.oasis-open.org/committees/xacml/.

[6] R. Bhatti, J. Joshi, E. Bertino, and A. Ghafoor, "Access Control in Dynamic XML-Based Web-Services with X-RBAC," in *Proceedings of the International Conference on Web Services (ICWS'03)*, 2003, pp. 243–249.

[7] C. A. Ardagna, E. Damiani, S. D. C. di Vimercati, and P. Samarati, "A Web Service Architecture for Enforcing Access Control Policies," *Electronic Notes Theoretical Computer Science*, vol. 142, pp. 47–62, 2006.

[8] J. Schlimmer, "Web Services Policy Framework (WS-Policy)," 2004, http://www-128.ibm.com/developerworks/webservices/library/specification/ws-polfram/.

[9] P. Nolan, "Understand WS-Policy processing," IBM Corporation, Tech. Rep., 2004.

[10] F. Paci, E. Bertino, and J. Crampton, "An Access-Control Framework for WS-BPEL," *International Journal of Web Services Research*, vol. 5, no. 3, pp. 20–43, 2008.

[11] A. Charfi and M. Mezini, "Aspect-Oriented Web Service Composition with AO4BPEL." in *ECOWS'04*, 2004.

[12] B. L. Mingjie Sun and P. Zhang, "Monitoring bpel-based web service composition using aop," in *Proceedings of The Eigth IEEE/ACIS International Conference on Computer and Information Science* , Washington, DC, USA, 2009.

[13] Y. Wu and P. Doshi, "Making bpel flexible - adapting in the context of coordination constraints using ws-bpel," in *Proceedings of the 2008 IEEE International Conference on Services Computing - Volume 1.*, Washington, DC, USA, 2008.

[14] M. Braem and D. Gheysels, "History-based aspect weaving for ws-bpel using padus," in *Proceedings of the Fifth European Conference on Web Services.*, Washington, DC, USA, 2007.

[15] J. Pavlich-Mariscal, L. Michel, and S. Demurjian, "Enhancing UML to Model Custom Security Aspects," in *Proceedings of the 11th International Workshop on Aspect-Oriented Modeling (AOM@AOSD'07)*, 2007.

[16] J. Evermann, "A Meta-Level Specification and Profile for AspectJ in UML," *Journal of Object Technology*, vol. 6, no. 7, pp. 27–49, 2007.

[17] G. Kiczales, E. Hilsdale, J. Hugunin, M. Kersten, J. Palm, and W. G. Griswold, "An Overview of AspectJ," in *Proceedings of the 15th European Conference on Object-Oriented Programming (ECOOP'01)*. London, UK: Springer-Verlag, 2001, pp. 327–353.

[18] V. Shah, "An aspect-oriented security assurance solution," Cigital Labs, Tech. Rep. AFRL-IF-RS-TR-2003-254, 2003.

[19] B. DeWin, "Engineering application level security through aspect oriented software development," Ph.D. dissertation, Katholieke Universiteit Leuven, 2004.

[20] A. Mourad, M.-A. Laverdière, and M. Debbabi, "Towards an aspect oriented approach for the security hardening of code," *Computers & Security*, vol. 27, no. 3-4, pp. 101–114, 2008.