ACSIJ Advances in Computer Science: an International Journal, Vol. 4, Issue 4, No.16 , July 2015
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

A comparative study and classification on web service security testing approaches

Azadeh Esfandyari

Department of Computer, Gilangharb branch, Islamic Azad University, Gilangharb, Iran
Azadeh.Esfandyari@gmail.com

Abstract

Web Services testing is essential to achieve the goal of scalable, robust and successful Web Services especially in business environment where maybe exist hundreds of Web Services working together. This Relatively new way of software development brings out new issues for Web Service testing to ensure the quality of service that are published, bound, invoked and integrated at runtime. Testing services poses new challenges to traditional testing approaches. Dynamic scenario of Service Oriented Architecture (SOA) is also altering the traditional view of security and causes new risks. The great importance of this field has attracted the attention of researchers. In this paper, in addition of presenting a survey and classification of the main existing web service testing approaches, web service security testing researches and their issues are investigated.

Keywords*: web service security testing, WSDL*

1. Introduction

The Web Services are modular, self-described and self-contained applications. With the open standards, Web Services enable developers to build applications based on any platform with any component modular and any programming language. More and more corporations now are exposing their information as Web Services and what's more, it is likely that Web Services are used in mission critical roles, therefore performance matters. Consumers of web services will want assurances that Web Services won't fail to return a response in a certain time period. So the Web Services testing is more important to meet the consumers' needs. Web Services' testing is different from traditional software testing. In addition, traditional testing process and tools do not work well for testing Web Services, and therefore, testing Web Services is difficult and poses many challenges to traditional testing approaches due to the above mentioned reason and mainly because Web Services are distributed applications with numerous runtime behaviors.

Generally, there are two kinds of Web Services, the Web Services are used in Intranet and the Web Services are used in Internet. Both of them face the security risk since message could be stolen, lost, or modified. The information protection is the complex of means directed on information safety assuring. In practice it should include maintenance of integrity, availability, confidentiality of the information and resources that used for data input, saving, processing and transferring [1]. To achieve reliable Web services, which can be integrated into compositions or consumed without any risk in an open network like the Internet, more and more software development companies rely on testing activities. In particular, security testing approaches help to detect vulnerabilities in Web services in order to make them trustworthy. The rest of paper is organized as follows: Section II presents an overview and a classification of web service testing approaches. Section III summarizes web service security testing approaches and issues. Finally, section IV gives a conclusion of the paper.

2. Overview and a classification of web service testing approaches

The Web Services world is moving fast, producing new specification all the time and different applications, and hence introducing more challenges to develop more adequate testing schemes. The challenges stem mainly from the fact that Web Services applications are distributed applications with runtime behaviors that differ from more traditional applications. In Web Services, there is a clear separation of roles between the users, the providers, the owners, and the developers of a service and the piece of software behind it. Thus, automated service discovery and ultra-late binding mean that the complete configuration of a system is known only at execution time, and this hinder integration testing [2]. To have an overview of web service testing approaches I use the classification proposed by [2]. But it seems that this classification is not sufficient for categorizing all existing approaches therefore new classification is introduced.

In [2] the existing web service testing approaches are classified to 4 classes by excluding the approaches that are based on formal method and data gathering:

- WSDL-Based Test Case Generation Approaches
- Mutation-Based Test Case Generation Approaches
- Test Modeling Approaches
- XML-Based Approaches

All Mutation-Based test case generation approaches that referred to in [2] like [3, 4] are based on WSDL and can placed in first class. Also there are approaches that in addition to considering WSDL specification use other scenarios to cope with limitation of WSDL specification

based test case generation so introduction new category is seemed necessary. The proposed classification is:

- WSDL-Based Test Case Generation Approaches
- Test Modeling Approaches
- XML-Based Approaches
- Extended Test Case Generation Approaches

2.1 WSDL-Based Test Case Generation Approaches

These approaches essentially present solution for generating test cases for web services based only on Web Services Description Language (WSDL).Research activities in this category are really extensive and not included in this paper. Two WSDL approaches is introduced in following.

Hanna and Munro in [5] present solution for test cases generation depending on a model for the XML schema datatypes of the input message parameters that can be found in WSDL specification of the Web Service under test. They consider the role of application builder and broker in testing web services. This framework use just boundary value testing techniques.

Mao in [6] propose two level testing framework for Web Service-based software. In service unit level, combinatorial testing method is used to ensure single service's reliability through extracting interface information from WSDL file. In system level, BPEL specification is converted into state diagram at first, and then state transition-based test cases generation algorithm is presented.
Obviously the researches that generate web service test case from WSDL by using various testing techniques like black box and random testing techniques and so on are placed in this category.

2.2 Test Modeling Approaches

Model-based testing is a kind of black-box testing, where these experiments are automatically generated from the formally described interface specification, and subsequently also automatically executed [7].
Frantzen et al. [7] discuss on a running example how coordination protocols may also serve as the input for Model-Based Testing of Web Services. They propose to use Symbolic Transition Systems and the underlying testing theory to approach modelling and testing the coordination.
Feudjio and Schieferdecker in [8] introduced the concept of test patterns as an attempt to apply the design pattern approach broadly applied in object-oriented software development to model-driven test development. Pattern driven test design effectively allows tests targeting semantical aspects which are highly critical for service availability testing, unlike other approaches that focus on syntactical correctness, to be designed at an early stage to drive the product development process and to help uncover failures prior to deployment of services [2].
Tsai et al. [9] present a Web Services testing approach based on a stochastic voting algorithm that votes on the outputs of the Web Service under test. The algorithm uses the idea of k-mean clustering to handle the multi-dimensional data with deviations. The heuristics is based on local optimization and may fail to find the global optimal results. Furthermore, the algorithm assumes that the allowed deviation is known, which may be hard to determine because the deviation is application dependent.

2.3 XML-Based Approaches

Tsai et al. [10] proposed an XML-based object-oriented (OO) testing framework to test
Web Services rapidly. They named their approach Coyote. It consists of two parts: test master and test engine. The test master allows testers to specify test scenarios and cases as well as various analyses such as dependency analysis, completeness and consistency, and converts WSDL specifications into test scenarios. The test engine interacts with the Web Services under test, and provides tracing information. The test master maps WSDL specifications into test scenarios, performs test scenarios and cases generation, performs dependency analysis, and completeness and consistency checking. A WSDL file contains the signatures specification of all the Web Services methods including method names, and input/output parameters, and the WSDL can be extended so that a variety of test techniques can be used to generate test cases. The test master extracts the interface information from the WSDL file and maps the signatures of Web Services into test scenarios. The test cases are generated from the test scenarios in the XML format which is interpreted by test engine in the second stage.
Di Penta et al. [11] proposed an approach to complement service descriptions with a facet providing test cases, in the form of XML-based functional and nonfunctional assertions. A facet is a (XML) document describing a particular property of a service, such as its WSDL interface. Facets to support service regression testing can either be produced manually by the service provider or by the tester, or can be generated from unit test cases of the system exposed as a service.

2.4 Extended Test Case Generation Approaches

Because of weak support of WSDL to web services semantical aspect some approaches don't confine themselves only to WSDL-Based Test Case Generation.

Damiani et al. [12] in order to guarantee the quality of the given services propose collaborative testing framework where different part participate in. They proposed a novel approach that uses a third party certifier as a trusted entity to perform all the needed test on behalf of the user and certify that a particular service has been tested successfully to satisfy the user's needs.

The open model scenario is a way to overcome the limitations of WSDL specification based test cases generation [12].

Since the service source code is generally not available, the certifier can gain a better understanding about the service behavior starting from its model. The benefit of such strategy is to allow the certifier to identify the critical areas of the service and therefore design test cases to check them [12].

2.5 Web service testing tools

Many tools have been implemented for testing Web Services. Next subsections describe briefly the three selected tools.

• SoapUI Tool

This tool is a Java based open source tool. It can work under any platform provided with Java Virtual Machine (JVM). The tool is implemented mainly to test Web Services such as SOAP, REST, HTTP, JMS and other based services. Although SoapUI concentrates on the functionality, it is also consider performance, interoperability, and regression testing [13].

• PushToTest Tool

One of the objectives of this open source tool is to support the reusability and sharing between people who are involved in software development through providing a robust testing environment. PushToTest primarily implemented for testing Service Oriented Architecture (SOA) Ajax, Web applications, Web Services, and many other applications. This tool adopts the methodology which is used in many reputed companies. The methodology consists of four steps: planning, functional test, load test, and result analysis. PushToTest can determine the performance of Web Services, and report the broken ones. Also, it is able to recommend some solutions to the problems of performance [14].

• WebInject Tool

This tool is used to test Web applications and services. It can report the testing results in real time, and monitor applications efficiently. Furthermore, the tool supports a set of multiple cases, and has the ability to analyze these cases in reasonable time. Practically, the tool is written in Perl, and works with the platforms which have Perl interpreter. The architecture of WebInject tool includes: WebInject Engine and Graphical User Interface (GUI), where the test cases are written in XML files and the results are shown in HTML and XML files [15].

3. Web service security testing overview and related work

Web services play an important role for the future of the Internet, for their flexibility, dynamicity, interoperability, and for the enhanced functionalities they support. The price we pay for such an increased convenience is the introduction of new security risks and threats, and the need of solutions that allow to select and compose services on the basis of their security properties [16]. This dynamic and evolving scenario is changing the traditional view of security and introduces new threats and risks for applications. As a consequence, there is the need of adapting current development, verification, validation, and certification techniques to the SOA vision [17].

To achieve reliable Web services, which can be integrated into compositions or consumed without any risk in an open network like the Internet, more and more software development companies rely on software engineering, on quality processes, and quite obviously on testing activities. In particular, security testing approaches help to detect vulnerabilities in Web services in order to make them trustworthy.

Concerning, the Web service security testing few dedicated works have been proposed. In [18], the passive method, based on a monitoring technique, aims to filter out the SOAP messages by detecting the malicious ones to improve the Web Service's availability. Mallouli et al. also proposed, in [19], a passive testing method which analyzes SOAP messages with XML sniffers to check whether a system respects a policy. In [20], a security testing method is described to test systems with timed security rules modelled with Nomad. The specification is augmented by means of specific algorithms for basic prohibition and obligation rules only. Then, test cases are generated with the "TestGenIF" tool. A Web Service is illustrated as an example. In [21] a security testing method dedicated for stateful Web Services is proposed. Security rules are defined with the Nomad language and are translated into test purposes. The specification is completed to take into account the SOAP environment while testing. Test cases are generated by means of a synchronous product between test purposes and the completed specification.

Some researchers (e.g., ANISETTI et al. [17]) focused on security certification. They believe that certification techniques can play a fundamental role in the service-based ecosystem. However, existing certification techniques are not well-suited to the service scenario: they usually consider static and monolithic software, provide certificates in the form of human-readable statements, and

consider systemwide certificates to be used at deployment and installation time. By contrast, in a service-based environment, we need a certification solution that can support the dynamic nature of services and can be integrated within the runtime service discovery, selection, and composition processes [22]

To certify that a given security property is holed by its service, two main types of certification processes are of interest: test-based certification and model-based certification.

According to Damiani et al. [23], test-based certification is a process producing evidence-based

Proofs that a (white- and/or black-box) test carried out on the software has given a certain result, which in turn shows that a given high-level security property holds for that software. Model-based certification can provide formal proofs based on an abstract model of the service (e.g., a set of logic formulas or a formal computational model such as a finite state automaton).

ANISETTI et al. [16] propose a test-based security certification scheme suitable for the service ecosystem. The scheme is based on the formal modeling of the service at different levels of granularity and provides a model-based testing approach used to produce the evidence that a given security property holds for the service. The proposed certification process is carried out collaboratively by three main parties: (i) a service provider that wants to certify its services; (ii) a certification authority managing the overall certification process; and (iii) a Lab accredited by the certification authority that carries out the property evaluation. Service model generated by the certification authority using the security property and the service specifications is defined at three level of granularity: WSDL-based model, WSCL-based model and implementation-based model. The certification authority sends the Service model together with the service implementation and the requested security property to the accredited Lab. The accredited Lab generates the evidence needed to certify the service on the basis of the model and security property and returns it to the certification authority. If the evidence is sufficient to prove the requested property the certification authority awards a certificate to the service, which includes the certified property, the service model, and the evidence. They also propose matching and comparison processes that return the ranking of services based on the assurance level provided by service certificates. Because of supporting the dynamic comparison and selection of functionally equivalent services, the solution can be easily integrated within a service-based infrastructure.

4. Conclusions

This paper had a review on main issue and related work on Web Service testing and Web Service security testing. Four classes are introduced for web service testing approaches. By considering this classification when security is concerned, the classes that are only based on WSDL specifications can't be useful for security testing. Since ignoring WSCL and implementation details doesn't allow the definition of accurate attack models and test cases. Because of the fourth class abstraction, it can include approaches that by complete modeling of service enable themselves to produce fine-grained test cases that will be used to certify the security property of service(e.g., ANISETTI et al. [16]) . Although some security concepts aren't taken into account in [16] (for instance reliability) and the level of complexity of its processes has increased, but it seems that this is the most comprehensive approach in web service security certification area.

5. Acknowledgements

The work reported in this paper was funded by Gilangharb branch, Islamic Azad University, Gilangharb, Iran.

References

[1] Li, Y., Li, M., & Yu, J. (2004). Web Services Testing, the Methodology, and the Implementation of the Automation-Testing Tool. In *Grid and Cooperative Computing* (pp. 940-947).

[2] Ladan, M. I. (2010). Web services testing approaches: A survey and a classification. In *Networked Digital Technologies* (pp. 70-79).

[3] Siblini, R., & Mansour, N. (2005). Testing web services. In *Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on* (p. 135).

[4] Andre, L., & Regina, S. (2009). V.: Mutation Based Testing of Web Services.*IEEE Software*.

[5] Hanna, S., & Munro, M. (2007, May). An approach for specification-based test case generation for Web services. In *Computer Systems and Applications, 2007. AICCSA'07. IEEE/ACS International Conference on* (pp. 16-23).

[6] Mao, C. (2009, August). A specification-based testing framework for web service-based software. In *Granular Computing, 2009, GRC'09. IEEE International Conference on* (pp. 440-443).

[7] Frantzen, L., Tretmans, J., & de Vries, R. (2006, May). Towards model-based testing of web services. In *International Workshop on Web Services–Modeling and Testing (WS-MaTe 2006)* (p. 67).

[8] Feudjio, A. G. V., & Schieferdecker, I. (2009). Availability testing for web services.

[9] Tsai, W. T., Zhang, D., Paul, R., & Chen, Y. (2005, September). Stochastic voting algorithms for Web services group testing. In *Quality Software, 2005.(QSIC 2005). Fifth International Conference on* (pp. 99-106).

[10] Tsai, W. T., Paul, R., Song, W., & Cao, Z. (2002). Coyote: An xml-based framework for web services testing. In *High Assurance Systems Engineering, 2002. Proceedings. 7th IEEE International Symposium on* (pp. 173-174).

[11] Di Penta, M., Bruno, M., Esposito, G., Mazza, V., & Canfora, G. (2007). Web services regression testing. In *Test and Analysis of web Services* (pp. 205-234).

[12] Damiani, E., El Ioini, N., Sillitti, A., & Succi, G. (2009, July). Ws-certificate. In*Services-I, 2009 World Conference on* (pp. 637-644).
[13] "SoapUI tool" , http://www.SoapUI.org.
[14] "PushToTest tool", http://www.PushToTest.com.
[15] "WebInject", http://www.WebInject.org/.

[16] Anisetti, M., Ardagna, C. A., Damiani, E., & Saonara, F. (2013). A test-based security certification scheme for web services. *ACM Transactions on the Web (TWEB)*, *7*(2), 5.
[17] Anisetti, M., Ardagna, C., & Damiani, E. (2011, July). Fine-grained modeling of web services for test-based security certification. In *Services Computing (SCC), 2011 IEEE International Conference on* (pp. 456-463).
[18] Gruschka, N., & Luttenberger, N. (2006). Protecting web services from dos attacks by soap message validation. In *Security and privacy in dynamic environments* (pp. 171-182).
[19] Mallouli, W., Bessayah, F., Cavalli, A., & Benameur, A. (2008, November). Security rules specification and analysis based on passive testing. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE* (pp. 1-6).
[20] Mallouli, W., Mammar, A., & Cavalli, A. (2009, December). A formal framework to integrate timed security rules within a TEFSM-based system specification. In*Software Engineering Conference, 2009. APSEC'09. Asia-Pacific* (pp. 489-496).
[21] Salva, S., Laurençot, P., & Rabhi, I. (2010, August). An approach dedicated for web service security testing. In *Software Engineering Advances (ICSEA), 2010 Fifth International Conference on* (pp. 494-500).

[22] Damiani, E., & Manã, A. (2009, November). Toward ws-certificate. In*Proceedings of the 2009 ACM workshop on Secure web services* (pp. 1-2).
[23] Damiani, E., Ardagna, C. A., & El Ioini, N. (2008). *Open source systems security certification*. Springer Science & Business Media.