International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

A Survey On XML-Injection Attack Detection Systems

Swati Ramesh Kesharwani[1] , Aarti Deshpande[2]

[1]M.E.Scholar in Computer Engineering, GHRCEM, Pune University, India

[2]Professor in Computer Engineering, GHRCEM, Pune University, India

Abstract: ***Web services are increasingly used as distributed systems on the Internet; they provide a standard means of interoperation among different software applications running on a variety of platforms and frameworks. However, the underlying technologies used by Web services, such as SOAP, HTTP, and XML, have fostered the deployment of well-known vulnerabilities in this new environment. This system specifically addresses XML injection attacks those that produce some change in the XML's internal components that aims to compromise the Web service application. This can be achieved by, for instance, injecting malicious content into an XML message, such as invalid XML characters. The classical detection system approach relies on building a signature-based database, cataloging attacks independently from each other. So, the proposed system is an XML injection strategy-based detection system, XHDS, to mitigate the time gap for 0-day attacks resulting from ontology's attack variations. Because many new and unknown attacks are derived from known strategies—considered signatures—low false-positive detection rates should occur. This project present XHDS as a hybrid approach that supports knowledge-based detection derived from a signature-based approach and then apply ontology to design the knowledge database for XML injection attacks against Web services.***

Keywords: Signature-based detection systems, Knowledge-based detection techniques, Web services, Ontology, XML Injection.

1. Introduction

Vulnerability detection tools are frequently considered the silver-bullet for detecting vulnerabilities in web services. However, research shows that the effectiveness of most of those tools is very low and that using the wrong tool may lead to the deployment of services with undetected vulnerabilities.[12] In this paper we propose a benchmarking approach to mitigate the time gap for 0-day attacks resulting from ontology's attack variations. This approach was used to define a concrete benchmark for XML Injection vulnerability detection system. One way to protect Web services from injection attacks is through signature-based detection systems. A signature is a payload that identifies an attack through some specific malicious context. Signature based detection systems usually lead to low software detection mistake rates namely, false-positive rates. However, one important limitation of signature-based attack detection is that it doesn't detect new unknown attacks, even if they have only small variations from a known payload. Another way to protect Web services from injection attacks is through knowledge-based detection systems, which apply protection based on some kind of previously known and cataloged behavior. Usually, two distinct classes are modeled: one for normal behavior, containing all expected actions that define such a profile, and another for attacks containing actions that aren't considered normal. Knowledge-based detection techniques can detect new attacks, but they usually produce a high false-positive rate.

Signature-based detection techniques are widely used, but they allow for 0-day attacks, which occur when vulnerability (software flaw) becomes publicly known before its fix is available. No matter how a vulnerability is disclosed to the world, a 0-day attack's effectiveness can vary from hours to months .The classical detection system approach relies on building a signature-based database, cataloging attacks independently from each other. It doesn't take into account whether an attack's strategy is similar for a set of existing attacks. Our goal in this work is to model attack elements as strategies—representing them by classes and relationships—in an ontology. Our belief is that by knowing the attack strategy, which defines a semantic relationship among attack elements, attack variations can be easily detected and automatically added to the ontology's database [11].

In this system, we describe an XML injection strategy-based detection system, XHDS, to mitigate the time gap for 0-day attacks resulting from ontology's attack variations. Because many new and unknown attacks are derived from known strategies—considered signatures—low false-positive detection rates should occur. This paper present XHDS as a hybrid approach that supports knowledge-based detection derived from a signature-based approach which apply an ontology to design the knowledge database for XML injection attacks against Web services.

2. Types of XML-Injection Attacks

1) XPath Injection
2) XQuery Injection
3) XSS Injection

1) XPath Injection:
XPath is a language solely used for selecting nodes from an XML document. XPath formats XML data as tree-structured values. XPath Injection attacks occur when a web site uses user-supplied information to construct an XPath query for XML data. By sending intentionally malformed information into the web site, an attacker can find out how the XML data is structured, or access data that he may not normally have access to. He may even be able to elevate his privileges on

the web site if the XML data is being used for authentication (such as an XML based user file).Querying XML is done with XPath, a type of simple descriptive statement that allows the XML query to locate a piece of information. Like SQL, you can specify certain attributes to find, and patterns to match. When using XML for a web site it is common to accept some form of input on the query string to identify the content to locate and display on the page. This input must be sanitized to verify that it doesn't mess up the XPath query and return the wrong data. XPath is a standard language; its notation/syntax is always implementation independent, which means the attack may be automated[10]. There are no different dialects as it takes place in requests to the SQL databases. Because there is no level access control it's possible to get the entire document.

2) XQuery Injection:
XQuery is a query and functional programming language that is designed to query and transform collections of structured and unstructured data, usually in the form of XML[10]. XQuery injection occurs when: Data enters a program from an untrusted source. The data used to dynamically construct an XQuery expression.

3) XSS Injection:
Cross-site scripting attacks may occur anywhere that possibly malicious users are allowed to post unregulated material to a trusted web site for the consumption of other valid users [10].The most common example can be found in bulletin-board web sites.

3. Classical Approaches for Detection Systems

- Signature based detection:
  Classical System uses Signature based detection technique. A signature is a payload that identifies an attack through some specific malicious context. Signature based detection systems usually lead to low software detection mistake rates namely, false-positive rates.
- Knowledge-based detection systems:
  Another way to protect Web services from injection attacks is through knowledge-based detection systems, which apply protection based on some kind of previously known and cataloged behavior. Usually, two distinct classes are modeled: one for normal behavior, containing all expected actions that define such a profile and another for attacks containing actions that aren't considered normal.

4. Existing Web Service Attack Detection Systems

Most proposals use classical approaches that don't work properly for Web services attack detection found in the technical literature. However, when an alternative approach for injection detection is used, it does not apply suitably to Web services attacks. The following approaches address some aspects related to our work.

Irfan Siddavatam and Jayant Gadge proposed running captured SOAP requests through a series of tests to determine if they could be classified as Web services attacks. Requests that fail the tests are filtered to be examined later.[1] The researchers presented tests and results but didn't detail the test detection mechanisms in their proposal.

Chan Yee and colleagues proposed an adaptable framework, applying agents, data mining, and fuzzy logic techniques to compensate for differences between anomaly and signature-based detection for Web services.[2] According to the authors, these techniques allow for decision making in uncertain and inaccurate environments, but no concrete results were provided.
Martin Bravenboer and his colleagues suggested using syntax

Embedding, according to the guest and host languages (such as XPath and Java).[3] The aim is to automatically generate code that will prevent vulnerabilities to injection attacks for example, adding character-escaping functions. The approach is generic and therefore can be applied to any language combination, but a limitation is the fact that not all languages have a context-free syntax.

Meisam Najjar and Mohammad Abdollahi Azgomi developed a hybrid Web services intrusion detection service to detect malicious behaviors in SOAP request/response messages.[4] The authors used misuse detection to verify an attack against a known set of signatures. Afterward, they use an anomaly-detection technique, based on normal profiles obtained through training, to alert unknown attacks. The authors mention that the proposal has an acceptable detection rate after a prototype implementation, but no actual test results were shown in the paper.

Zhichun Li and his colleagues proposed an approach to mitigate the 0-day attack problem through an automated signature-generation system, focusing on polymorphic worms.[5] The system generated signatures based on invariant worm contents and on small content variations. Tests showed that the approach is viable both in performance and accuracy for the proposal's scope, but the article didn't describe Web services attacks.

Jeffrey Undercoffer and his colleagues applied an ontology to model attack classes mainly on the basis of the attack target, while also considering the attack's means and consequences.[6] The proposed ontology was shared by several attack detection systems with the objective of disseminating new attacks acknowledged by any of them. This approach, however, didn't use axioms or inference, and no tests were presented in their work.

Zakaria Maamar and colleagues presented an ontology-based approach to specify and secure contexts of Web services.[7] Each context in the ontology had specific encryption/decryption mechanisms to authenticate messages that were sent and received. The authors didn't model attacks in their ontology; they mainly focused on representing the context of Web services composition and security needs.

Artem Vorobiev and Jun Han proposed the closest approach to the work we present in this article; they applied an

ontology to specifically address the Web services attacks domain.[8] However, the ontology implementation wasn't included in the research, and the proposal didn't use inference to detect unknown attacks; it mostly used ontologies to represent a common vocabulary.

5. Design of Proposed System

Proposed System uses Knowledge based detection system knowledge-based detection derived from a signature-based approach..To protect Web services from injection attacks is through knowledge-based detection systems, which apply protection based on some kind of previously known and cataloged behavior. Usually, two distinct classes are modeled: one for normal behavior, containing all expected actions that define such a profile, and another for attacks containing actions that aren't considered normal. Knowledge-based detection techniques can detect new attacks, but they usually produce a high false-positive rate. Signature-based detection techniques are widely used, but they allow for 0-day attacks, which occur when vulnerability (software flaw) becomes publicly known before its fix is available. No matter how vulnerability is disclosed to the world, a 0-day attack's effectiveness can vary from hours to months. The classical detection system approach relies on building a signature-based database, cataloging attacks independently from each other. It doesn't take into account whether an attack's strategy is similar for a set of existing attacks.[11]

Our goal in this work is to model attack elements as strategies representing them by classes and relationships in ontology. Our belief is that by knowing the attack strategy, which defines a semantic relationship among attack elements, attack variations can be easily detected and automatically added to the ontology's database.

It describe an XML injection strategy-based detection system, XHDS, to mitigate the time gap for 0-day attacks resulting from an ontology's attack variations. Because many new and unknown attacks are derived from known strategies—considered signatures—low false-positive detection rates should occur. We present XHDS as a hybrid approach that supports knowledge-based detection derived from a signature-based approach. We then apply an ontology to design the knowledge database for XML injection attacks against Web services.

6. Measures

It is very difficult to compare the effectiveness of method that implement different vulnerability detection approaches, based on the number of vulnerabilities reported for the same piece of code. This way, our proposal is to characterize vulnerability detection method using the F-Measure proposed by van Rijsbergen, which is largely independent of the way vulnerabilities are counted. In fact, it represents the harmonic mean of two very popular measures (precision and recall), which, in the context of vulnerability detection, can be defined as:

Precision: a ratio of correctly detected vulnerabilities to the number of all detected vulnerabilities. In our context it can be represented as follows:

$$precision = \frac{TP}{TP + FP}$$

Recall: a ratio of correctly detected vulnerabilities to the number of known vulnerabilities. In our context,

It can be represented as follows:

$$recall = \frac{TP}{TV}$$

Where:

- TP (true positives) is the number of true vulnerabilities detected (i.e., vulnerabilities that, in fact, exist in the code);
- FP (fault positives) is the number of vulnerabilities detected that, in fact, do not exist.
- TV (true vulnerabilities) is the total number of vulnerabilities that exist in the workload code.

Assuming an equal weight for precision and recall, the formula for the ***F-Measure*** is:

$$F - Measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

A highly effective system will generate a high F-Measure (which obviously ranges between 0 and 1)[12].

7. Conclusion and Future Work

A Web services administrator can consider the informative messages for investigation, aiming to tune the detection system, ensuring that the proposal approach doesn't produce false positives during detection. The 0-day attacks are eliminated for the inferred instances because a new attack instance is automatically added to the ontology's knowledge database at the moment its first detected. In future work, we intend to extend the ontology to contemplate other attacks that burden Web services, such as denial of service. As the number of attack classes and axioms grows, so does the inference power of our hybrid approach

References

[1] Siddavatam and J. Gadge, "Comprehensive Test Mechanism to Detect Attack on Web Services," Proc. 16th IEEE Int'l Conf. Networks (ICON 08), IEEE, 2008, pp. 1–6.
[2] C.G. Yee, W.H. Shin, and G.S.V.R.K. Rao, "An Adaptive Intrusion Detection and Prevention (ID/IP) Framework for Web Services,"Proc. Int'l Conf. Convergence Information Technology, IEEE, 2007, pp. 528–534.
[3] M. Bravenboer, E. Dolstra, and E. Visser, "Preventing Injection Attacks with Syntax Embeddings," Science of Computer Programming, vol. 75, no. 7, 2010, pp. 473–495.

[4] M.S.A. Najjar and M.A. Azgomi, "A Distributed Multi-approach Intrusion Detection System for Web Services," Proc. 3rd Int'l Conf.Security of Information and Networks (SIN 10), ACM, 2010, pp.238–244.
[5] Z. Li et al., "Hamsa: Fast Signature Generation for Zero- Day Polymorphic Worms with Provable Attack Resilience," Proc. 2006 IEEE Symp. Security and Privacy (SP 06), IEEE CS, 2006, pp. 32–47 .
[6] J. Undercoffer et al., "A Target-Centric Ontology for Intrusion Detection," Proc. IJCAI-03 Workshop Ontologies and Distributed Systems, Morgan Kaufmann, 2004, pp. 47–58.
[7] Z. Maamar, N.C. Narendra, and S. Sattanathan, "Towards an Ontology-Based Approach for Specifying and Securing Web Services," Information and Software Technology, Elsevier, 2005, pp. 441–455.
[8] Vorobiev and J. Han, "Security Attack Ontology for Web Services,"Proc. 2nd Int'l Conf. Semantics, Knowledge, and Grid (SKG 06), IEEE CS, 2006, p. 42.
[9] Morin et al., "A Logic-Based Model to Support Alert correlation in Intrusion Detection," Information Fusion,vol. 10, no. 4, 2009, pp. 285–299
[10] Booth et al., "Web Services Architecture," working group note, W3C, Feb. 2004; www.w3.org/TR/ws-arch.2.34
[11] Thiago Mattos Rosa, Altair Olivo Santin and Andreia Malucelli,"Mitigating XML Injection 0-Day Attacks through Strategy-Based Detection Systems" Copublished by the IEEE Computer and Reliability Societies 1540-7993/2013 IEEE, July/August 2013 .
[12] N. Antunes and M. Vieira, "Benchmarking Vulnerability Detection Tools for Web Services," Proc. IEEE Int'l Conf.Web Services (ICWS), IEEE CS, 2010; doi;10.1109/ICWS.2010.76

Author Profile

Swati Ramesh Kesharwani received received Bachelor of Computer Science Engineering from Kavikulguru Institute of Technology and Science, Nagpur University. She is now pursuing Master of Computer Engineering, G. H. Raisoni College of Engineering and Management, University of Pune, India. Her research interest is Web Security.

Aarti Deshpande, Professor, Department of Computer Engineering, G. H. Raisoni College of Engineering and Management, University of Pune, India.