Penetration Test Tool for XML-based Web Services

Christian Mainka* Vladislav Mladenov† Juraj Somorovsky* Jörg Schwenk

Horst Görtz Institute for IT-Security, Ruhr-University Bochum, Germany
{christian.mainka, vladislav.mladenov, juraj.somorovsky, joerg.schwenk}@rub.de

Abstract

XML is a platform-independent data format applied in a vast number of applications. Starting with configuration files, up to office documents, web applications and web services, this technology adopted numerous – mostly complex – extension specifications. As a consequence, a completely new attack scenario has raised by abusing weaknesses of XML-specific features.

In the world of web applications, the security evaluation can be assured by the use of different penetration test tools. Nevertheless, compared to prominent attacks such as SQL-Injection or Cross-site scripting (XSS), there is currently no penetration test tool that is capable of analyzing the security of XML interfaces. In this paper we motivate for development of such a tool and describe the basic principles behind the first automated penetration test tool for XML-based web services named *WS-Attacker*.

Keywords: Penetration Test Tool, Web Service, XML Security, Signature Wrapping, Single Sign-On, WS-Attacker

1 Introduction

Service Oriented Architectures (SOAs) found the main idea for well-known and wide-spread technologies like Cloud computing and can be found in military services, e-Government, as well as in private and enterprise solutions. The main advantage of SOA is software reuse, modularization, and service out-sourcing. For its realization, well known interfaces have to be defined and the eXtensible Markup Language (XML) has become one of the key technologies for this task. Surrounded with the related W3C-standards such as SOAP, Web Services Description Language (WSDL) and XML Schema, XML is more than just a simple data description language – it is a full-featured platform-independent markup language.

The need for flexible security mechanisms in such architectures led to the development of addtional standards for securing SOA protocols. WS-Security relies on the already existing standards *XML Encryption* and *XML Signature* and applies them to SOAP-based web services. Moreover, WS-Trust is for establishing trust domains and WS-Policy/WS-SecurityPolicy are responsible for creating policies between communicating parties. Apart from web services, the Security Assertion Markup Language (SAML) OASIS Standard has gained increased popularity for Single Sign-On scenarios in enterprise web applications.

* The authors were supported by the Sec2 project of the German Federal Ministry of Education and Research (BMBF, FKZ: 01BY1030).

† The authors were supported by the SkIDentity project of the German Federal Ministry of Economics and Technology (BMWi, FKZ: 01MD11030).

In: A. Editor, B. Coeditor (eds.): Proceedings of the XYZ Workshop, Location, Country, DD-MMM-YYYY, published at http://ceur-ws.org

Unfortunately, due to the complex design of these standards (e.g. XPath, XSLT, XML Signature, XML Encryption), their implementation has become very difficult. As a result of this, a lot of highly critical security flaws could be found in the processing of XML Signatures on SAML-based Single Sign-On frameworks [SMS$^+$12]: eleven out of 14 systems were vulnerable to the XML Signature Wrapping (XSW) attack which was published by McIntosh and Austel seven years ago [MA05]. In the context of web services, a further work showed the effectiveness of this attack by breaking the Amazon EC2 as well as the Eucalyptus Cloud web interfaces [SHJ$^+$11]. Even the confidentiality of XML Encryption protected messages could be annuled. Due to a bad usage of the CBC mode, the symmetric XML Encryption could be broken [JS11] and by applying Bleichenbacher' attack technique, the same authors also broke the asymmetric encryption [JSS12].

Besides the attacks on cryptographic primitives, there are also very efficient Denial-of-Service (DoS) attacks which abuse XML-specific characteristics. One example for this is the HashDoS attack which constructs special formed XML code in order to store XML attributes or namespace declarations in the same bucket of a vulnerable hash table and thus enormously slows down its processing[1]. Another example known as XML bomb uses XML entity declarations in a recursive way so that a message consisting of only a few KB will be expanded to several GB [JGHL07].

A huge problem from the security point of view is the complexity of the existing XML standards, which are often misunderstood. As a result of this, they are often not able to identify XML-specific security risks and therefore can not fix them. In the area of penetration testing tools for web applications customers can nowadays choose between several automated tools (or single components of such) for analyzing the security of systems in general or scanning for specific vulnerabilities, e.g. XSS and SQL-Injection. However, currently there is no known (commercial or open-source) software on the market that offers the ability to search and identify XML-specific weaknesses. This is our motivation to start working on a penetration test tool for web services.

2 Foundations

2.1 Attacking Web Services

The basic idea of a web service is to define an interface for message communication. The internal web service logic extracts the necessary information and forwards it to the underlying back-end. The problem of this approach is that the used XML standards for defining such an interface are very powerful and complex, thus a web service has mainly two different threat models:

Non-specific XML attacks abuse weaknesses in the back-end of an application, e.g Buffer Overflows or SQL-Injection.

Specific XML attacks exploit vulnerabilities in SOAP/web service and XML. They attack the XML parsing mechanism to enforce a DoS or build unexpected SOAP messages, e.g. change the SOAPAction header to confuse the web service logic.

It is important to mention that non-specific attacks are well known from web applications. However, compared to attacks such as XSS and SLQ-Injection, XML-specific attacks are totally new. They provoke the web service interface to behave unexpectedly by using XML-specific features. Currently, some penetration testing tools are able to handle web services, e.g. SOAP Sonar by Crosschecknetworks[2] or WSFuzzer by OWASP[3]. These tools support attacks such as SQL-Injection or XPath-Injection. Nevertheless, they do not handle all the XML-specific attacks. Therefore, we decided to develop our own penetration test tool called *WS-Attacker*[4] in order to fill the gap [MSS12].

2.2 XML Specific Attacks

A lot of XML-specific attacks exist and are known for a long time. Table 1 gives an overview on currently published attacks mainly taken from [JGH09]. Their classification, detailed information and even more attacks can be found on our website[5]. Due to the limited space, the next section will only focus on the XSW attack.

[1] CVE-2012-0841: `https://bugzilla.redhat.com/show_bug.cgiid=CVE-2012-0841`

[2] `http://www.crosschecknet.com/products/soapsonar.php`

[3] `https://www.owasp.org/index.php/Category:OWASP_WSFuzzer_Project`

[4] `http://sourceforge.net/projects/ws-attacker/`

[5] `http://ws-attacks.org`

XML Signature Wrapping	Attack on XML Encryption	Oversize Payload
Coercive Parsing	SOAPAction spoofing	XML Injection
WSDL Scanning	Metadata spoofing	Attack Obfuscation
Oversized Cryptography	BPEL State Deviation	Instantiation Flooding
Indirect Flooding	WS-Addressing spoofing	Middleware Hijacking

Table 1: Overview of existing XML-specific attack attacks.

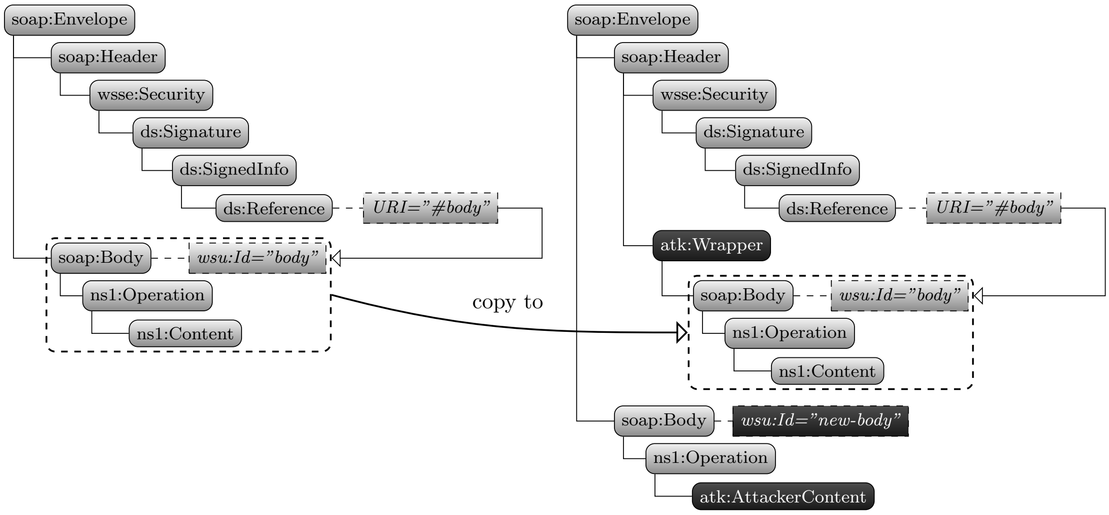

Figure 1: Basic XML Signature Wrapping scenario.

2.3 XML Signature Wrapping

XML Signature Wrapping (XSW) is an XML-specific attack first published by McIntosh and Austel in 2005 [MA05]. The very basic attack concept is shown in Figure 1.

Generally, the attack stems from the fact that the XML processing logic is mostly divided into two components: signature verification logic and application logic. The task of the signature verification logic is only to verify the signed content. In the depicted figure, the signature verification logic detects the signed content by only looking for any ID attribute with a specific value: `wsu:Id="body"`. After applying the attack as shown, it can still find the signed element in the attacker message, but it does not notice that it has moved. The application logic instead determines the element to process by just using the first element found as a child of the `<soap:Body>` element and ignores the ID attribute. Thus, the attacker's content is executed.

Note that different more complex attacks of this type exist [SMS+12, SHJ+11].

3 WS-Attacker's Task and XSW Attack's Complexity

The vast number of attacks on XML-based systems and the lack of an existing penetration test tool motivated us to develop *WS-Attacker*. The goal was to create a software solution which can be easily extended with any kind of XML-specific attacks. It is simple to use even for non-XML Security experts – which is realized by a easily understandable GUI which can be configured with only a few clicks – and can help to detect XML-specific vulnerabilities. Therefore, the user has to (1) load a WSDL, which identifies the web service endpoint, (2) send a test-request to the server to learn its *normal state* (behavior on untapered requets), (3) select the attack plugins, and (4) press a start button.

The need for such a penetration test tool is founded in the complexity of the attacks. Looking back to the XSW attack mentioned in the previous section as an example, Figure 2 visualizes its complexity. It is possible to have a large number of signed elements and each of it can be wrapped into a couple of positions within the XML document, e.g. located *somewhere* in the `<soap:Header/>`, or in the `<soap:Body/>`. Additionally, the wrapper can be placed as the first child, the last child, or somewhere in between. For each of this position, there can be additional adjustments (e.g. change the ID-value or keep it). The XSW attack can become even more complex

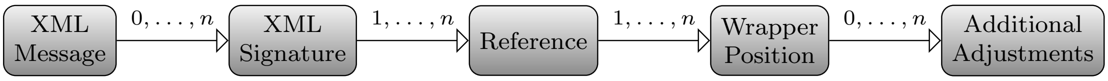

Figure 2: The complexity of the XSW attack.

when taking care of XPath based signatures [GJLS09] or the namespace injection technique [JLS09]. As a result of the different attack variants, a human attacker is not able to test all attack vectors.

This workflow clarifies that the attack performation by hand is nearly impossible. Besides incredible time consumption as a result of the different attack variants, a human attacker is not able to test all attack vectors.

Note that this is only an example for XSW attack, but this or a similar complexity can also be found on attacks on XML Encryption and XML DoS.

4 Future Work

In this section we give an overview of the known attacks on web service, which could be used to extend our framework.

4.1 XML-Specific Attacks

Our framework currently covers only a few of the attacks shown in Table 1. At the moment, there are already some existing attacks implemented, e.g. SOAPAction Spoofing and WS-Addresing Spoofing[6]. Even the powerful XSW attack can be automatically performed, including all attack variants and wrapping possibilities on ID-based signatures as well as on XPath-based systems. SAML over SOAP is also already implemented, and we are currently focused on browser-based SAML Single Sign-On as an extension of the WS-Attacker. However, the implementation of this extension is not trivial at all. Besides the XSW-attacks we want to integrate further tests regarding the configuration of the provider and already known bugs. Therefore, we need a very flexible and extensible software architecture able to generate dynamically SAML tokens. Furthermore, we require an evaluation logic analyzing the reaction of the tested system in response to the applied attack vectors. However, this evaluation is not a trivial issue due to the differences between the various systems accepting SAML tokens. Additionally, we are close before the release of XML-specific DoS attacks. The attacks on XML Encryption or the XXE (Xml eXternal Entity) attacks[7] are considered as our future work.

4.2 Beyond XML

Besides the XML-based services and protocols, other standards such as OpenID or OAuth became increasingly important in Single Sign-On scenarios. Moreover, current researches show the expanding usage of OpenID[8]. In addition to SAML, OpenID and OAuth are the most used protocols in the Cloud environment in order to authenticate users. For this reason their security became a part of common researches and has already been investigated by Wang et al. [WCW12]. They found critical security bugs in the authentication process, which allowed them to sign-in as an arbitrary user by misusing control flaws between Service Providers and Identity Providers like Facebook and Google. This work has been complemented by Sun and Beznosov [SB12]. However, none of these studies explicitly handles signature processing flaws at the Identity Providers. Thus, we see the automatic testing of OpenID and OAuth signature validation as a challenge in our future work, which could be included in our WS-Attacker framework.

In addition to the SOAP-based web service standards, many REST[9]-based web service interfaces support custom XML-based security mechanisms or follow the newest JSON security standards: JSON Web Signature [JRH12] and JSON Web Encryption [JBS12]. Jager et al. have already shown that their attacks on XML Encryption [JSS12] could be directly applied to the JSON Web Encryption standard. Automation and extension of these attacks could be considered as a next part of our future work.

[6] http://ws-attacks.org
[7] http://www.agarri.fr/blog
[8] http://trends.builtwith.com/docinfo/OpenID
[9] Representational state transfer

5 Conclusion

The threat of XML-based attacks has significantly increased. So does their application field: Besides web services, also Single Sign-On systems are attackable as latest researches have revealed [SMS+12]. This underlines the necessity of an automatic penetration test tool. Our solution – WS-Attacker – currently supports the first XML-specific attacks on web services, including the powerful XSW attacks with the majority of the known attack variants.

This paper gave an overview of the WS-Attacker framework and its basic functionalities. It sketched the future directions in the development of further XML-specific attacks, as well as of attacks beyond XML and web services. We believe that such an all-in-one solution will significantly help developers in finding vulnerabilities in their systems.

References

[GJLS09] Sebastian Gajek, Meiko Jensen, Lijun Liao, and Jörg Schwenk. Analysis of signature wrapping attacks and countermeasures. In *ICWS*, pages 575–582. IEEE, 2009.

[JBS12] M. Jones, J. Bradley, and N. Sakimura. JSON Web Signature (JWS) – draft-ietf-jose-json-web-signature-06, October 2012.

[JGH09] Meiko Jensen, Nils Gruschka, and Ralph Herkenhöner. A survey of attacks on web services. *Computer Science - R&D*, 24(4):185–197, 2009.

[JGHL07] Meiko Jensen, Nils Gruschka, Ralph Herkenhner, and Norbert Luttenberger. Soa and web services: New technologies, new standards - new attacks. In *Proceedings of the 5th IEEE European Conference on Web Services (ECOWS)*, 2007.

[JLS09] Meiko Jensen, Lijun Liao, and Jörg Schwenk. The curse of namespaces in the domain of xml signature. In Ernesto Damiani, Seth Proctor, and Anoop Singhal, editors, *SWS*, pages 29–36. ACM, 2009.

[JRH12] M. Jones, E. Rescorla, and J. Hildebrand. JSON Web Encryption (JWE) – draft-ietf-jose-json-web-encryption-06, October 2012.

[JS11] Tibor Jager and Juraj Somorovsky. How To Break XML Encryption. In *The 18th ACM Conference on Computer and Communications Security (CCS)*, October 2011.

[JSS12] Tibor Jager, Sebastian Schinzel, and Juraj Somorovsky. Bleichenbacher's attack strikes again: breaking PKCS#1 v1.5 in XML Encryption. In Sara Foresti and Moti Yung, editors, *ESORICS*, LNCS. Springer, 2012.

[MA05] Michael McIntosh and Paula Austel. XML signature element wrapping attacks and countermeasures. In *SWS '05: Proceedings of the 2005 Workshop on Secure Web Services*, pages 20–27, New York, NY, USA, 2005. ACM Press.

[MSS12] Christian Mainka, Juraj Somorovsky, and Jörg Schwenk. Penetration testing tool for web services security. In *SERVICES Workshop on Security and Privacy Engineering*, June 2012.

[SB12] San-Tsai Sun and Konstantin Beznosov. The devil is in the (implementation) details: an empirical analysis of oauth sso systems. In *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12, pages 378–390, New York, NY, USA, 2012. ACM.

[SHJ+11] Juraj Somorovsky, Mario Heiderich, Meiko Jensen, Jörg Schwenk, Nils Gruschka, and Luigi Lo Iacono. All Your Clouds are Belong to us – Security Analysis of Cloud Management Interfaces. In *The ACM Cloud Computing Security Workshop (CCSW)*, October 2011.

[SMS+12] Juraj Somorovsky, Andreas Mayer, Jörg Schwenk, Marco Kampmann, and Meiko Jensen. On breaking saml: Be whoever you want to be. In *21st USENIX Security Symposium*, Bellevue, WA, August 2012.

[WCW12] Rui Wang, Shuo Chen, and XiaoFeng Wang. Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services. In *IEEE Symposium on Security and Privacy (Oakland), IEEE Computer Society*, May 2012.