

# XML 重写攻击检测技术研究

刘宝龙, 杨 威, 陈 桦

(西安工业大学 计算机科学与工程学院, 陕西 西安 710021)

**摘 要:**细粒度的 XML 数字签名中存在重写攻击的问题, 已有多种方案用来检测 XML 重写攻击。文中在分析评估了这些检测方案的基础上, 讨论了针对各种常见重写攻击类型的安全应对方案以及各种检测方案的最佳应用场景。研究结果表明: 安全策略、验证互补(过滤器)、FastXPath 以及标记 DOM 树中元素位置方案能有效地检测到常见的重写攻击方式, 且除内联法及验证互补(位置指示器)方案外, 已有方案都可应用于有效的检测中间人攻击和重放攻击的场景中。然而, 针对修改签名元素上下文关联信息的攻击方式, 已有方案都不能检测到。

**关键词:**XML 重写攻击; 安全策略; SOAP Account; 验证互补; FastXPath; 重定向攻击; 多 Security 头攻击

中图分类号: TP309.2

文献标识码: A

文章编号: 1673-629X(2016)06-0101-05

doi: 10.3969/j.issn.1673-629X.2016.06.022

## Study on Detecting Technique for XML Rewriting Attack

LIU Bao-long, YANG Wei, CHEN Hua

(School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710021, China)

**Abstract:** There is rewriting attack problem in the fine-grained XML digital signature now. There are several countermeasures can be used to detect XML rewriting attack. It makes a discussion on the security scheme to deal with the common rewriting attacks and the best application scenarios of the existing detection scheme based on the analysis and evaluation of the existing detection scheme. The study results show that security policy, verification complementary (filter), FastXPath and mark element position scheme in the DOM tree can detect the common attacks effectively and existing scheme can apply to detecting man-in-the-middle attack and replay attack effectively except for inline approach and verification complementary (position indicator) scheme. However, for attacks against modifying signature element context-sensitive information, all the existing detection scheme can't detect.

**Key words:** XML rewriting attack; security policy; SOAP Account; verification complementary; FastXPath; redirection attack; multiple security header attack

## 0 引言

随着 Web 服务的迅速发展和广泛应用, 其通信安全问题日益重要。Web 服务间的通信是使用基于 XML 格式的 SOAP 消息实现的。XML 数字签名可以对 SOAP 消息的特定部分签名, 实现细粒度的签名, 这也导致了 XML 重写攻击问题的出现, 进而威胁到 SOAP 消息的传输安全。2005 年, McIntosh 和 Austel 首次提出了 XML 重写攻击<sup>[1]</sup>。XML 重写攻击是指 Web 服务中对 SOAP 消息的恶意拦截、操作和传输的总称。目前, 已有多种检测 XML 重写攻击的方案, 它们分别能检测到某些攻击类型, 但是各自并不能检测到所有的攻击类型。文中在分析评估了这些检测方案

的基础上, 讨论了针对各种常见攻击的安全应对方案以及每种方案的最佳应用场景, 为 Web 服务中针对 XML 数字签名的应用提供了参考。

## 1 已有的 XML 重写攻击检测方案

### 1.1 安全策略

Web 服务安全策略用来描述 Web 服务的安全能力或者安全需求。WS-Policy<sup>[2]</sup>为描述 Web 服务的安全策略提供了一个通用的策略框架。WS-SecurityPolicy<sup>[3]</sup>定义了用于描述各种安全需求使用的安全策略断言集。Web 服务安全策略中通过安全策略断言来指定 SOAP 消息的签名部分以及签名消息使用的安全令

收稿日期: 2015-09-05

修回日期: 2015-12-10

网络出版时间: 2016-05-25

基金项目: 教育部留学归国人员科研启动金资助项目(2013693); 面向重大装备和能源化工的制造业信息化综合应用示范项目(2012BAF12B04); 陕西省教育专项科研计划项目(15KJ1350)

作者简介: 刘宝龙(1976-), 男, 副教授, 博士, 研究方向为 XML 安全技术。

网络出版地址: <http://www.cnki.net/kcms/detail/61.1450.TP.20160525.1700.020.html>

牌,以此来保护 SOAP 消息的完整性。

在执行安全通信前,服务方基于 WS-Security 规范的安全机制定制自己的安全需求,在 WS-Policy 策略框架下,使用 WS-SecurityPolicy 定义的断言集描述安全需求并生成安全策略文件,再通过 WS-PolicyAttachment 机制实现安全策略文件与 Web 服务的绑定。请求方获取服务方的安全策略文件,根据其中的安全需求对 SOAP 消息进行安全处理,并将处理后的消息发送给服务方。服务方根据安全策略文件对其进行安全验证,根据验证结果来决定是否建立整个通信过程。

由于策略文件的复杂性,Web 服务所需的安全策略可能并不完善。微软的 SAMOA 项目研究人员提出基于 TulaFale 形式化验证<sup>[4]</sup>以及 WSE policy advisor<sup>[5]</sup>工具来检验安全策略文件中是否存在 XML 重写攻击漏洞,使策略文件的作者可以根据漏洞修正安全策略,以提高策略文件的安全可靠性。

## 1.2 内联法

(1) SOAP Account 方法。

SOAP Account 方法<sup>[6-9]</sup>通过添加一个新的 SOAP Account 元素来标记 SOAP 消息的结构信息,这个元素在消息创建时被添加进来。

SOAP Account 元素中包含的 SOAP 结构信息如下:

- Envelope 元素的子元素个数;
- SOAP Header 子元素的个数;
- 签名中引用的个数;
- 每个签名对象的后继和前驱关系。

(2)改进的 SOAP Account 方法。

Benameur 等对 SOAP Account 方法进行了改进<sup>[10]</sup>,修改了 SOAP Account 元素中标记的结构信息。

改进的 SOAP Account 方法中,SOAP Account 元素包含的 SOAP 信息如下:

- 标记签名元素的层次信息;
- 标记签名元素的父节点名称;
- 为签名元素的父节点添加一个“ID”属性,使它唯一地标记签名元素的父节点信息。

内联法中,SOAP Account 元素在 SOAP 消息发送之前添加到消息的 Header 元素中。发送方必须对 SOAP Account 元素进行签名。在 SOAP 消息传输时,可能经过一个或者多个中介体,中介体可以对 SOAP 消息进行处理并添加自己的 SOAP Account,但是每一个后继中介体必须签名自己的 SOAP Account、前一个签名节点的签名信息以及自身需要签名的内容。接收方对签名消息进行验证,并计算 SOAP 消息的 SOAP Account 信息,将计算结果和接收的 SOAP Account 元素标记的信息进行比较,如果不匹配,则 SOAP 消息可

能受到了 XML 重写攻击。

## 1.3 验证互补方案

该方案中提出修改签名验证函数,在签名验证有效的情况下,返回一个过滤器或者位置指示器<sup>[11]</sup>。这样,可以通过以下两种方式检测 XML 重写攻击。

(1)签名验证函数返回一个过滤器。XML 数字签名实际上是对签名对象的摘要值进行签名。过滤器的功能是只将 Reference 引用中进行摘要值计算的数据传送给应用逻辑。这样,只有签名消息被传送给应用逻辑,应用逻辑也就无法访问那些可能包括攻击信息的未签名消息。因此,应用逻辑处理的一定是签名消息。

(2)签名验证函数返回一个位置指示器。通过位置指示器,应用逻辑可以获取 SOAP 消息中使用绝对 XPath 表达式表示的签名节点的绝对路径信息,并将它和消息发送前的绝对路径信息进行比较,如果不一致,则消息可能受到了攻击。

## 1.4 FastXPath 方案

该方案中使用 FastXPath 表达式对签名对象进行引用,以标记文档根节点到签名节点的绝对路径信息,通过固定签名节点的位置,来实现检测 XML 重写攻击的目的<sup>[12]</sup>。

FastXPath 表达式是指不包含通配符的绝对 XPath 表达式。它不仅标记了从文档根节点到签名节点路径上的每一个节点名称,还标记了签名节点的层次信息、父节点信息及在兄弟节点中的位置信息,严格限制了攻击者修改文档中签名节点位置的灵活性。这样,基于修改签名位置信息重写攻击便可被检测出来。

## 1.5 标记 DOM (Document Object Model) 树中元素位置

该方法是在 SOAP 消息头中添加一个新的 <SignedElements>头元素,来标记 SOAP 消息的 DOM 树中 Envelope、Body 和签名元素的位置信息<sup>[13]</sup>。

在 SOAP 消息发送之前,发送方根据后序遍历算法对 SOAP 消息的 DOM 树进行编码,获得 SOAP 元素在树中的位置标记,并在 SOAP 消息头中创建 <SignedElements>头元素,其中包括 Envelope、Body 和签名元素的位置标记。<SignedElements>头元素自身必须被签名。接收方通过验证 SOAP 消息生成的 DOM 树中 Envelope、Body 以及签名元素的位置标记和 <SignedElements>元素中 Envelope、Body 以及签名元素的位置标记是否一致来检验 SOAP 消息是否受到了 XML 重写攻击。

## 2 针对常见攻击的安全应对方案

XML 重写攻击中常见的攻击方式有:封装签名消

息添加到 SOAP 头中的攻击、重定向攻击、多 Security 头攻击、修改上下文关联信息的攻击。本节在分析评估了已有检测方案的基础上,讨论了针对各种常见攻击的安全应对方案,为有效检测 XML 重写攻击提供理论参考。

## 2.1 封装签名消息添加到 SOAP 头中的攻击

封装签名消息添加到 SOAP 头中的攻击是指攻击者将签名消息封装到伪造的头元素里并添加到 SOAP 消息头中,使其不被接收端的应用逻辑处理,并在原签名消息的位置添加自己的攻击信息,使应用逻辑处理添加的攻击信息。

安全策略方案中可指定使用绝对 XPath 表达式表示签名元素的绝对路径信息,而该种类型的攻击会改变签名元素的绝对路径信息,接收方在根据安全策略文件中的绝对路径信息指定的元素进行验证时,会出现摘要值的不匹配,导致签名验证失败。

验证互补(过滤器)方案中,由于过滤器只把签名消息传送给应用逻辑,应用逻辑处理的只能是签名消息,而不会是攻击信息。因此,此方案下的签名不受该种攻击类型的影响。验证互补(位置指示器)方案中,接收方可以将通过位置指示器获取的当前 SOAP 消息中签名元素的绝对路径信息和消息发送前的绝对路径信息进行比较来发现该种攻击。

由于该种类型的攻击必然会改变签名节点的位置,而在根据 FastXPath 表达式对签名对象进行引用验证时,会出现摘要值的不匹配,导致签名验证失败。因此,FastXPath 方案可以检测到。

该种类型的攻击会改变 DOM 树中 Envelope、Body 或者签名元素的位置标记。所以,标记 DOM 树中元素位置的方案可以通过这种改变检测出该种攻击。

SOAP Account 方法能检测到将伪造的头元素添加到 Envelope 或 Header 元素中的攻击,但不能检测出将整个 Envelope 元素封装后添加到 Header 子元素中的攻击,因为这时 SOAP Account 中标记的结构信息并未改变。因此,不能确定 SOAP Account 方法对该种攻击类型的检测结果。

改进的 SOAP Account 方法中,攻击者如果只是封装签名消息添加到 SOAP 头中,必然会导致签名元素的父节点信息的改变,虽然层次信息在极少的情况下可能保持不变,但是能检测到这种攻击类型。攻击者如果封装签名消息及其父节点并添加到 SOAP 头中,在层次信息可能保持不变的极少情况下,不能检测到这种攻击类型。所以,不能确定对该种攻击类型的检测结果,但是它能检测到大规模的该种攻击类型。

因此,针对封装签名消息添加到 SOAP 头中的攻击,安全策略、验证互补、FastXPath 和标记 DOM 树中

元素位置方案能够检测到,可作为针对该种攻击类型的安全应对方案。但是,由于不能确定内联法的检测结果,故内联法不能作为针对该种攻击类型的安全应对方案。

## 2.2 重定向攻击

重定向攻击是指攻击者将 SOAP 消息的回复地址 <ReplyTo>元素封装到伪造的头元素里,使其不被接收端的应用逻辑处理,实现修改 SOAP 消息回复地址的攻击行为。假设 Header 中签名的 <ReplyTo>头元素为可选元素,接收方并不确定发送方是否指定了回复地址。这里主要是为了区别 2.1 节的安全策略,如果 <ReplyTo>头元素声明为必须元素,2.1 节的安全策略就能检测到。

安全策略文件中可通过绝对 XPath 表达式“/Envelope/Header/ReplyTo”对 <ReplyTo>元素进行签名引用(区别于 2.1 节的安全策略)。攻击者如果将 <ReplyTo>元素封装到伪造的元素 <Wrapper>里,在进行签名验证时,签名引用中的绝对 XPath 表达式会指向一个空的节点集,必然出现摘要值的不匹配,导致签名验证失败。

验证互补(过滤器)方案中,过滤器只将签名信息传送给应用逻辑,应用逻辑处理的只能是签名信息。所以,该方案不受该种攻击类型的影响。验证互补(位置指示器)方案中,接收方可以将通过位置指示器获取的当前 SOAP 消息中签名元素的绝对路径信息和消息发送前的绝对路径信息进行比较来发现该种攻击。

由于该种类型的攻击会改变签名节点的绝对路径信息,使用 FastXPath 表达式对签名消息进行引用验证时,会指向一个空的节点集,出现摘要值的不匹配,导致签名验证失败。因此,FastXPath 方案可以检测到。

该种攻击会改变 DOM 树中 Envelope、Body 或者签名元素的位置标记。所以,标记 DOM 树中元素位置的方案可以通过这种改变检测出该种攻击类型。

SOAP Account 方法中,攻击者如果只是封装 <ReplyTo>元素并添加到 <ReplyTo>元素原来的位置,会改变 <ReplyTo>元素的前驱和后继关系,导致 SOAP Account 元素标记的结构信息的改变。这时,能检测到。如果攻击者将 <ReplyTo>元素的父元素封装到伪造的头元素 <Wrapper>里添加在 <ReplyTo>元素原来的位置,SOAP Account 元素标记的结构信息并不会改变。这时,不能检测到。所以,不能确定 SOAP Account 方法的检测结果。

改进的 SOAP Account 方法中,攻击者如果封装 <ReplyTo>元素或者其父元素添加到 SOAP 头中 <ReplyTo>元素原来的位置,必然会导致签名元素的父节

点信息或者层次信息的改变。这时,能检测到该种攻击类型。然而,攻击者如果封装<ReplyTo>元素及其父元素并添加到 SOAP 体中,同时保持<ReplyTo>元素的层次信息不变。这时,不能检测到该种攻击类型。所以,不能确定对该种攻击类型的检测结果,但是它能检测到大规模的该种攻击。

因此,针对重定向攻击,安全策略、验证互补、FastXPath 和标记 DOM 树中元素位置方案能检测到,可作为针对该种攻击类型的安全应对方案。但是,由于不能确定内联法的检测结果,故不能作为针对该种攻击类型的安全应对方案。

### 2.3 多 Security 头攻击

SOAP 消息中通过 Header 头中的消息标识符<MessageID>和 Security 头中的时间戳<Timestamp>元素来防止重放攻击,但是消息标识符和时间戳自身可能会被攻击者修改<sup>[14]</sup>。多 Security 头攻击就是指攻击者将原 Security 头中的时间戳<Timestamp>元素封装在一个新的 Security 头中并添加到 Header 头中的攻击(这里以时间戳<Timestamp>元素为例)。消息标识符<MessageID>和时间戳<Timestamp>元素如果一个被修改,攻击者就可以实现对 SOAP 消息的重放攻击。针对<MessageID>元素的攻击,情况同 2.2 节。这里主要讨论检测时间戳<Timestamp>元素的攻击。

安全策略文件中通过绝对的 XPath 路径表达式“/Envelope/Header/Security/Timestamp”对<Timestamp>元素进行签名引用,但是多 Security 头攻击并不会改变<Timestamp>元素的绝对路径,这时可通过为签名元素的父节点 Security 添加一个 role 属性值.../ultimateReceiver,使用 XPath 表达式“/Envelope/Header/Security[@ role = “.../ultimateReceiver”]/Timestamp”来对签名对象进行引用,使签名元素的父节点 Security 必须被接收端处理。攻击者如果修改 Timestamp 元素的位置,会使 XPath 表达式指向一个空的节点集,导致签名验证失败。

由于该种攻击类型必然会改变 SOAP Header 的子元素数目以及兄弟元素之间的关系,所以,内联法能检测到此种类型的攻击。

验证互补(过滤器)方案中,过滤器只将签名消息传送给应用逻辑,应用逻辑处理的只能是签名消息,而不会是攻击信息。所以,不受该种攻击类型的影响。

FastXPath 方法中使用 FastXPath 表达式“/Envelope/Header/Security [ @ role = “.../ultimateReceiver”]/Timestamp”来对 Timestamp 元素进行签名引用,而该种攻击类型必然会改变签名节点的位置,使 FastXPath 表达式指向一个空的节点集,导致签名验证失败。所以,可以检测到该种攻击类型。

该种类型的攻击必然会改变 DOM 树中 Envelope、Body 或者签名元素的位置标记。所以,标记 DOM 树中元素位置方案可以通过这种改变检测出该种攻击。

验证互补(位置指示器)方案中,由于此类攻击中,攻击者将签名消息封装到一个新的 Security 头中并添加到 Header 头中,并未改变签名节点到文档根节点的绝对路径信息。这样,攻击前后的绝对路径信息并未变化,接收方就不能通过绝对路径信息的变化来检测出该种攻击。

因此,针对多 Security 头攻击,安全策略、内联法、验证互补(过滤器)、FastXPath 和标记 DOM 树中元素位置方案能够检测到,可作为针对该种攻击类型的安全应对方案。但是,验证互补(位置指示器)方案不能检测到,故不可作为针对该种攻击类型的安全应对方案。

### 2.4 修改上下文关联信息的攻击

修改上下文关联信息的攻击是指攻击者修改签名消息的上下文关联信息,改变签名消息的原始语义,从而达到篡改签名的目的。上下文关联信息是指与签名消息在语义上相关的上下文关联元素的内容。

安全策略中通常使用绝对的 XPath 表达式指定签名对象或者对签名对象进行引用,来保护签名元素的位置信息,但是该种攻击类型并不会改变签名元素的位置信息,只是改变了签名消息的上下文关联元素的内容以及签名的原始语义,而根据安全策略并不能检测到这些变化。因此,不能检测到该种攻击。

由于该种攻击类型只是修改签名元素的上下文关联元素的内容,并不会改变 SOAP Account 元素中标记的任何结构信息。所以,内联法不能检测到。

攻击者如果修改签名消息的上下文关联元素来实现重写攻击,表明应用逻辑需要处理未签名的内容,而验证互补(过滤器)方案只会将签名消息传送给应用逻辑,显然该方案不能满足应用逻辑的要求。所以,这里认为该方案不能检测到该种攻击。验证互补(位置指示器)中,应用逻辑通过指示器获取 SOAP 消息中签名元素的绝对路径信息,但是该种攻击并不会导致绝对路径信息的变化,所以不能通过路径信息的变化检测出该种攻击。

该种攻击类型并不会改变签名节点的位置信息以及 DOM 树中 Envelope、Body 或者签名元素的位置标记。所以,FastXPath 以及标记 DOM 树中元素位置的方案都不能检测出该种攻击。

因此,针对修改签名信息上下文关联信息的攻击,安全策略、内联法、验证互补、FastXPath 和标记 DOM 树中元素位置方案都不能检测到,故均不能作为针对该种攻击类型的安全应对方案。

## 2.5 应用场景分析

重放攻击是指攻击者将窃取的 SOAP 消息重复发送给接收方,以实现某种非法的目的。SOAP 消息中通过消息标识符<MessageID>元素和时间戳<Timestamp>元素来防止重放攻击,如果有一个元素被修改,就可能受到了重放攻击。所以,阻止重放攻击就是保护<MessageID>和<Timestamp>元素不被攻击。<MessageID>和<Timestamp>元素可能受到的攻击类型同重定向攻击和多 Security 头攻击。如果某个方案能检测到这两种攻击类型,就可以实现重放攻击的检测。

中间人攻击是指攻击者将截获的 SOAP 消息修改后再转发给接收方。常见的攻击方式有:封装签名消息添加到 SOAP 头中,在原签名消息位置添加自己的攻击信息,使应用逻辑处理自己的攻击信息;改变消息的请求或者回复地址,也就是重定向攻击。如果能检测到这两种攻击,就能有效地检测中间人攻击。

分析讨论结果表明:已有方案都能应用于有效地检测中间人攻击的场景中。除内联法及验证互补(位置指示器)外,其他方案均能应用于有效地检测重放攻击的场景中。而对于修改上下文关联信息的攻击方式,已有的方案都不能检测到。

## 3 结束语

文中分析讨论了针对各种常见 XML 重写攻击类型的安全应对方案以及每种方案的应用场景。对于封装签名消息添加到 SOAP 头中的攻击、重定向攻击、多 Security 头攻击,这些检测方案都能或部分检测到。但是,对于修改上下文关联信息的攻击,已有方案都不能检测到。为了增强针对 XML 重写攻击的检测能力,还需展开进一步的研究。

## 参考文献:

- [1] McIntosh M, Austel P. XML signature element wrapping attacks and countermeasures [C]//Proceedings of the 2005 ACM workshop on secure web services. Fairfax, USA: ACM Press, 2005: 20-27.
- [2] Web Services Policy 1.2 - Framework (WS-Policy) [EB/OL]. 2006-05-25. <http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/>.
- [3] Della-Libera G, Hondo M, Janczuk T, et al. Web Services Security Policy language (WS-Security Policy) [EB/OL]. 2002. <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/wssecuritypolicy.asp>.
- [4] Bhargavan K, Fournet C, Gordon A, et al. TulaFale: a security tool for web services [C]//Proceedings of the 2nd international symposium on formal methods for components and objects. [s. l.]: [s. n.], 2004: 197-222.
- [5] Bhargavan K, Fournet C, Gordon A. An advisor for web services security policies [C]//Proceeding of the secure web services workshop. [s. l.]: [s. n.], 2005: 1-9.
- [6] 朱云, 徐枫, 宴轲. 基于 XML 重写的 SOAP 安全 [J]. 信息工程大学学报, 2013, 14(5): 634-640.
- [7] Rahaman M A, Marten R, Schaad A. An inline approach for secure soap requests and early validation [C]//Proceeding of the Open Web Application Security Project Europe conference (OWASP). Leuven, Belgium: [s. n.], 2006: 1-15.
- [8] Rahaman M A, Schaad A. Soap-based secure conversation and collaboration [C]//Proceedings of IEEE international conference on web services. Salt Lake City, Utah: IEEE, 2007: 471-480.
- [9] Rahaman M A, Schaad A, Rits M. Towards secure soap message exchange in a SOA [C]//Proceedings of the 3rd ACM workshop on secure web services. Virginia, USA: ACM, 2006: 77-84.
- [10] Benameur A, Kadir F A, Fenet S. XML rewriting attacks: existing solutions and their limitations [C]//Proceeding of the international conference on applied computing. Algarve, Portugal: [s. n.], 2008: 94-102.
- [11] Gajek S, Liao L J, Schwenk J. Breaking and fixing the inline approach [C]//Proceedings of the 2007 ACM workshop on secure web services. New York, NY, USA: ACM, 2007: 37-43.
- [12] Gajek S, Jensen M, Schwenk J. Analysis of signature wrapping attacks and countermeasures [C]//Proceedings of IEEE international conference on web services. [s. l.]: IEEE, 2009: 575-582.
- [13] Barhoom T S, Rasheed R S K. Position of signed element for SOAP message integrity [J]. Journal of Computer Information Systems, 2011, 2(1): 21-25.
- [14] Kadir F A. RewritingHealer: an approach for securing web service communication [D]. Sweden: KTH Royal Institute of Technology, 2008.