文章编号:1001 - 9081(2005)02 - 0407 - 02

基于 XSLT 的 XML 元素级加密技术及其应用

杨广铭,张志浩
(同济大学 教育技术与计算中心,上海 200092)
(memphis_young@ hotmail. com)

摘 要:XML 元素级加密是指对 XML 文档中的某一特定部分进行加密。在讨论基于 XSLT 的元素级加密技术的基础之上,提出了使用扩展函数来实现 XML 元素级加密/解密的功能,并通过具体的例子来说明其可行性。相对于其他元素级加密技术,该方法最大的优点就是避免了部署与维护的麻烦,它把所有的安全操作都限制在了样式表内,而不涉及 XSLT 处理器,具有高可用性。

关键词:XML; XSLT; 元素级加密; 扩展函数; .NET
中图分类号: TP393.08 **文献标识码**:A

Technology of XML element-wise encryption based on XSLT and its application

YANG Guang-ming, ZHANG Zhi-hao
(Computer Center, Tongji University, Shanghai 200092, China)

Abstract: XML element-wise encryption is used to encrypt a certain part of XML document. In this paper, auther discussed the technology of element-wise encryption based on XSLT, and introduced a method that extension functions were used to provide function of encryption and decryption with some examples. Compared with other element-wise encryption technologies, the biggest advantage of this one is to avoid deployment and maintenance. All the security operations are confined to stylesheet and not involved in XSLT processor. So this technology has high-usability.

Key words: XML; XSLT; element-wise encryption; extension function; .NET

0 引言

借助于 Internet,XML 提供一个在异构平台间交换信息的简便方法,它能充分利用现有的各种通信协议(如 HTTP 和 SSL 等),并且可以十分容易地穿过防火墙,因此,XML 文档被广泛用作商业信息的载体。但是由于 XML 遵循一种与系统无关的字符集(Unicode),所以它是可读的,而 Internet 作为一个公共网络,通常对于敏感数据的非授权访问和攻击缺乏保护,这就要求在传输过程中对 XML 文档进行加密。虽然在大多数情况下,一些标准的加密协议如 IPSec 和 SSL 都能很好地提供传输过程中 XML 文档的保密性,而安全邮件如 PGP (Pretty Good Privacy)和 S/MIME 则能保证 XML 文档接收后被保存在文件系统中的安全性。但是这些方法通常都是针对整个 XML 文档进行加密,而在有些情况下只需对 XML 文档中的某一特定部分进行加密,Maruyama 和 Imamura 首先提出了元素级加密(Element-Wise Encryption)的概念,如下述几种情况就需要使用元素级加密:

1) 在一个三方协议中,中间商不能访问敏感元素的内容。

2) 当一个访问控制策略要求 XML 文档的特定部分只对特权用户可读时。

3) 当非敏感内容需要针对特定接收者进行修改时。

使用元素级加密可以使得加密后的 XML 文档和其解密的形式保持相同的组织结构,因此它能被应用程序透明地处理,而不须考虑它是否被加密过。在这类 XML 中元素值与文档结构相分离,例如在一个接收并转发 XML 文档的应用中,转发控制只与一个特定元素的值有关,而与其他所有元素都无关。实现元素级加密最简单有效的方法就是利用 XML 处理器,这种方法最大的优点是构造比较容易,可以用任何编程语言来实现专用 XML 处理器,但缺点是需要在系统中每一个处理 XML 文档安全的节点上部署和维护这一专用软件。W3C 的 XML Encryption 正是使用这一方法,在 XML 处理器中包含了安全的考虑,通过 XML 文档中的属性进行控制(用于初始化和调用的目的)。IBM 的 XML Security Suite 就实现了这种处理器。

1 XSLT 实现元素级加密

XSLT(eXtensible Stylesheet Language Transformation)是由 W3C 定义的一种对 XML 进行转换的语言。它的出现使得构建 XML 文档或对文档中的元素进行运算变得更加方便快捷,同时还提供了高级声明编程语言的所有好处。如果把加密和解密看作是另一种 XML 的转换操作,即从加密的 XML 转换

收稿日期:2004 - 07 - 15

作者简介:杨广铭(1979 -),男,上海人,硕士研究生,主要研究方向:网络技术、信息管理; 张志浩(1942 -),男,上海人,博士生导师,主要研究方向:计算机网络、数据库.

到未加密的 XML,反之亦然,那么 XSLT 将能很好的完成这一工作。使用 XSLT 进行元素级 XML 加密可以很好地避免上述方法中部署和维护专用 XML 处理器的麻烦。因为它只需要一个标准的 XSLT 处理器,所有 XML 的安全操作都将被限制在样式表中,而不涉及 XSLT 处理器。

一个使用 XSLT 实现的 XML 加密/解密过程如图 1 所示。

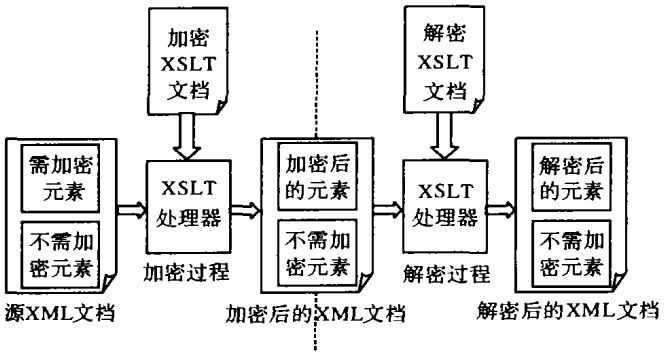

图 1 基于 XSLT 的 XML 元素级加密和解密的过程

在图 1 中一个源 XML 文档包含需加密的元素和不需加密的元素两个部分,经过一个转换(包含在加密 XSLT 文档中)产生一个加密的 XML 文档,转换只针对需加密的元素。第二个样式表(解密 XSLT 文档)将这个加密的 XML 转换成它原来的形式或者其他用户所需要的形式。

完全遵循 XSLT 语法来构建加密和解密的样式表看起来似乎是可行的,但是由于 XSLT 本身缺乏赋值运算符,这就使得即使在样式表中编写最简单的程序也会增加它的复杂性,更不用说编写复杂的加密算法了。因此使用"纯"的 XSLT 来进行加密/解密这一方法并不高效。

为了既提供传统编程语言的功能,又不致于影响样式表的复杂度,扩展函数(extension object)的概念被提了出来。在 XSLT1.0 规范中并没有提供扩展函数,但是,由于现在绝大多数的 XSLT 处理器都利用 Microsoft 的 COM 或者 Java 实现了某种形式的脚本扩展,为了顺应这一形势,目前的 XSLT1.1 提案中加入了对扩展函数的支持。

显然,在本文所讨论的 XSLT 中需要两种不同扩展函数:用作加密库与样式表之间接口的函数和用于解密库与样式表之间接口的函数,使用这些扩展函数都是为了能更好地进行元素级加密。一个实现 XML 加密或解密的样式表从概念上来说由三部分组成:

1) 用于匹配 XML 文档中对应标签的模板声明。这样就可以保留原 XML 文档中所包含的文档结构。为了能在输出文档中保留原文档的结构,每一个标签都需要在样式表中有一个相匹配的模板以便进行复制。根据 W3C 的建议,XSLT 处理器将默认地丢弃那些没有匹配的标签,这是我们所不愿看到的;

2) 扩展函数。用于对 XML 中的文本字符串进行加密与解密;

3) 属性识别与复制。一些包含在源 XML 文档中的属性要求被复制到输出文档中,而另一些则要求被舍弃。这也是通过模板匹配来实现。

使用包含扩展函数的 XSLT 进行 XML 文档加密/解密的框架如图 2 所示。

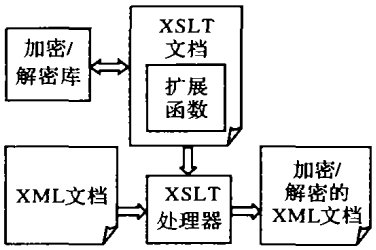

图 2 使用扩展函数的 XSLT 加密/解密框架

在这个框架中 XSLT 处理器接受一个样式表和一个源 XML 文档作为输入,并利用样式表中的扩展函数与外部的加密/解密库进行交互,以生成加密/解密的目标 XML 文档。

2 应用实例

本文以一个电子商务为例说明如何使用 XSLT 实现 XML 加密。问题描述如下:客户在网上商店购买商品时,将生成一份元素级加密的 XML 订单,该订单包含所购商品的信息和采用发卡银行公钥加密后的信用卡信息。接着,订单会被发送给指定的商家,商家在确认商品信息后会再次将文档发送给银行,银行用私钥解密并确认客户信用卡信息后给商家一个结算,这样商家就可以发货给客户了。

2.1 应用环境

在这个应用中选用的 XSLT 处理器是 Microsoft. NET 框架的 XSLT 实现,在. NET 框架中实现 XSLT 处理器的是 XslTransform 类。. NET 中的 XSLT 处理器与 Microsoft 传统的 MSXML 组件在功能上相类似,它们都对 XSLT 和 XPath 规范进行了扩展,其中最重要的扩展就是对扩展元素的支持。它们提供了一个顶层扩展元素 < msxsl:script > (元素 < xsl:stylesheet > 和 < xsl:transform > 的子元素)用以在样式表中包含扩展函数。

但是. NET 的 XSLT 处理器相对于 MSXML 最大的区别在于它可以用任何. NET 支持的语言(包括 C#, Visual Basic, Jscript. NET)在样式表中编写扩展函数,而 MSXML 目前仅支持 VBScript 和 JavaScript。本案例中我们选用 C#作为扩展函数的编写语言。

2.2 设计与实现

本部分将给出这个例子加密实现的步骤,解密实现与加密相类似。

客户在网上商店购买商品后,网站会自动生成一份 XML 形式的订单。由于订单中包含了应该对商家不可见的客户信用卡号(< cardno > 元素),所以在将订单发往商家前需先对卡号进行加密。XslTransform 先使用自己的成员函数 Load 读取一个加密 XSLT,在这个 XSLT 中包含了用于加密的扩展函数 Encrypt,该函数使用 RSA 算法进行加密,加密使用的是发卡银行的公钥。然后再用成员函数 Transform 对该 XML 订单进行转换。加密后得到 < cardno > bIuu/DktWwXMV0dqMVZue7WSFpnTTI/8 </cardno >,其中的 BASE64 编码即为加密的信用卡号,其他元素不变。

(下转第 411 页)

由上述的实验可以得出,当网络中发生基于 UDP 协议的 DDoS 攻击时,其网络流量依然保持了自相似特性,而且攻击流量本身也具有自相似特性。无论是单节点的拒绝服务攻击还是多节点的协同的分布式拒绝服务攻击,无论是恒定攻击流量还是周期性变化的攻击流量都保持了自相似的特性。实验所用的协议之所以是 UDP 协议是因为从流量变化的角度看,相对于 TCP 和 ICMP 协议,UDP 协议比较简单,只有攻击流而没有响应流,能更好的反映出流量本身的特性。当网络流量大时,TCP 协议本身就有其拥塞控制机制来调节流量(虽然这种调节机制对于 DDoS 来讲是微不足道),而 ICMP 协议也有对等于攻击流的响应流在链路中。

实际的攻击流往往是这三种协议混合而成的,所以还有很多的问题有待研究,如当 TCP 拥塞机制作用时的流量变化、进出流量是否具有自相似特性,大规模网络的攻击流自相似特性等,都是下一步要做的工作。

参考文献:

[1] LELAND WE, TAQQU MS, WILLINGER W, *et al.* On the self-similarity nature of Ethernet traffic[A], in Proc ACM SIGCOMM [C], 1993.183 – 193.

[2] PAXSON V, FLOYD S. Wide-Area Traffic: The failure of Poisson modeling[J]. in IEEE/ACM Trans on Networking, 1995,3(3):226 – 44.

[3] CROVELLA ME, BESTAVROS A. Self-similarity in World Wide Web traffic: Evidence and possible cause[J]. in IEEE/ACM Trans on Networking, 1977,6:835 – 846.

[4] POPESCU A. Traffic Self-Similarity[A]. (invited) tutorial, IEEE International Conference on Telecommunications, ICT2001 [C]. Jun,2001.

[5] PARK K, KIM G, CROVELLA M. On the effect of self – similarity on network performance[A]. in Proc of the SPIE International Conf on Performance and Control of Network System[C], 1997.296 – 310.

[6] OST A, BOUDEWIJN RH. Modeling and Evaluation of pseudo self-similar traffic with Infinite-State Stochastic Petri Nets[A]. in Proc of the workshop on formal method and telecom[C], 1999.120 – 136.

[7] PEHA JM. Protocols can make traffic appear self - similar [A]. in Proc of the 1997 IEEE/ACM/SCS Comm. Networks and Distributed System. Modeling and Simulation Conf[C], 1997. 47 – 52.

[8] VERES A, BODA M. The chaotic nature of TCP congestion control [A]. in Proc IEEE INFOCOM 2000, Tel Aviv, Israel[C], 2000. 1715 – 1723.

[9] 林原. 基于网络自相似性的 DDOS 攻击检测[D]. 成都: 电子科技大学,2002.

[10] HU G, DOLZER K, GAUGER CM. Does Burst Assembly Really Reduce the Self-Similarity[J]. Conference on Optical Fiber Communication(OFC2003). Technical Digest Series, 2003,86:124 – 126.

[11] PAXON V. Fast Approximation of Self-Similar Network Traffic[R]. Technical Report LBL36750, U of California, Berkeley, Apr 1995.

[12] The Internet Traffic Archive[EB/OL]. http://ita.ee.lbl.gov/html/contrib/BC.html, 1989 – 10 – 03.

(上接第 408 页)

订单 XML:

```
<order>
   <name>Yang</name>
   <product>Computer</product>
   <price>5999</price>  <quantity>1</quantity>
   <cardno>5555 – 1234 – 4567 – 6789</cardno>
</order>
```

加密 XSLT 代码片断:

```
<xsl: stylesheetversion = "1.0"
   xmlns: xsl = "http://www.w3.org/1999/XSL/Transform"
   xmlns: msxsl = "urn: schemas – microsoft – com: xslt"
   xmlns: encryption = "urn: the – xml – encryption: xslt – csharp" >
    < msxsl: script language = " C #" implement – prefix = "
        encryption" >
        <![CDATA[
        public string Encrypt(string s)
        { ..... }
        ]] >
    </msxsl: script >
    .....
    <xsl: template match = "cardno" >
       <xsl: copy >
          <xsl: value – of select = "encryption: Encrypt(text()) "/ >
       </xsl: copy >
    </xsl: template >
</xsl: stylesheet >
```

3　结语

尽管 XSLT 有处理 XML 文档的结构和内容的能力,但是如果要其提供诸如加密/解密等其他附加功能则会增加样式表的复杂性并且限制的加密文档所能使用的方法。通过使用扩展函数可以很好地弥补 XSLT 的这一弱点。由此可见,使用基于扩展函数的 XSLT 来对 XML 文档进行元素级加密是有实际意义的。

参考文献:

[1] W3C. Extensible Markup Language (XML) 1.0 (Third Edition); W3C Recommendation 04 February 2004[S/OL]. http://www.w3.org/TR/2004/REC – xml – 20040204, 2004 – 02.

[2] MARUYAMA H, IMAMURA T. Element - Wise XML Encryption [EB/OL]. http://lists.w3.org/Archives/Public/xml-encryption/2000Apr/att-0005/01-xmlenc.html, 2000 – 04.

[3] W3C. XSL Transformations (XSLT) Version 1.0; W3C Recommendation 16 November 1999[S/OL]. http://www.w3.org/TR/1999/REC-xslt-19991116, 1999 – 11.

[4] W3C. XML Encryption Syntax and Processing; W3C Recommendation 10 December 2002[S/OL]. http://www.w3.org/TR/2002/REC-xmlenc-core-20021210, 2002 – 12.

[5] BARTLETT RG, COOK MW. XML Security Using XSLT[R]. Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03), 2003.1.