Web服务安全性的研究

王翠茹　许正伟　袁和金　马慧敏

（华北电力大学计算机科学与技术学院　保定071003）

摘　要　随着Internet的蓬勃发展，Web服务技术作为一种新的Web应用程序的分支炙手可热，对传统的软件设计产生了巨大的影响，但是对于Web服务的安全领域的研究却不成熟。本文指出了现有安全技术保护Web服务存在的缺点和不足，结合Web服务技术新的安全性需求，提出了基于消息层处理的Web服务安全性模型，并详细介绍了消息处理层的实现方法。采用该模型实现的Web服务安全系统能够提供各种类型的安全服务，具有安全性、可扩展性、服务的透明性、安全的可控性、消息级端对端的安全性等诸多优点。

关键词　Web服务，Web服务安全，消息层

Research of Web Services Security

WANG Cui-Ru　XU Zheng-Wei　YUAN He-Jin　MA Hui-Min

(School of Computer Science and Technology, North China Electric Power University, Baoding 071003)

Abstract　Along with the development of Internet, Web services technology is a new branch of Web application program, and it has become a hotspot in computer science. However, it has not made great progress in research on Web services security. This paper points out the deficiencies and shortcoming of Web services security. Combined with its new security requirement, this paper puts forward a Web services security model based on message layer. The realization method of message processing layer is introduced in detail. The implementation version of the model can provide various security services, and has advantages such as security, scalability, security controllability and end-to-end security in message level.

Keywords　Web services, Web services security, Message layer

1　引言

Web服务是一种基于XML的革新技术，但是，Web服务的安全领域却是有待开发的大西部。现在的状况是：基于Web服务的软件开发十分简单，但实现安全可靠的Web服务却非常困难。因此，设计与实现安全的Web服务系统成为当务之急[1]。

2　现有安全技术保护Web服务的不足

现有的安全技术在保护Web信息安全方面已经十分成熟，主要有：SSL（安全套接字层）、VPN（虚拟专用网）、防火墙技术和HTTP认证。然而，Web服务作为一项新技术，有其特殊的安全需求，所以现有的各项技术保护Web服务时都存在不足[1,2]。

SSL(Security Socket Layer)是目前Web上具有真正意义上的安全通讯标准，但是在性能、中间节点和选择性保护等问题上存在着一定的缺点；使用VPN可以很好地保护Web服务，但是不可能要求所有的人都使用VPN，此外，建设VPN也增加了系统的开销；使用防火墙可以屏蔽非法的IP地址的访问，但有时也限制了合法用户的访问，此外，它不能实现其它安全服务，如数字签名和加密；现有的安全认证机制种类繁多，一般位于不同的协议层，应用时往往需要复杂的配置，不同的软件环境的情况更为复杂。

总之，Web服务技术提出的新的安全性要求是：选择性加密、端对端的安全、基于应用层的安全性和集成于Web服务体系的安全。现有的安全技术显然无法满足这些要求。

3　基于消息层的Web服务安全性模型

3.1　模型的提出

针对于Web服务安全性要求的特点，结合现有安全模型，本文提出了基于消息层的Web服务安全的系统模型。图1给出了该模型的结构。

从图1可以看出，在Web服务器端和服务的客户端，该模型分别增加了一个消息处理层。与此同时，增加了安全所需的相应组件，如数字证书库、用户数据库等。下面详细介绍各层的功能[3~5]。

(1)服务客户层

该层是Web服务的客户层，它主要有两方面的功能：当客户请求服务时，将请求信息串行化为XML格式的SOAP消息，发送给下一个处理层。当接收到相应消息时，并行化消息为相应的对象，传递

王翠茹　教授，硕士生导师，主研方向为数据库与信息管理系统。袁和金　讲师，主研方向为数据库与信息管理系统。许正伟　硕士研究生，主研方向为数据库与信息管理系统。马慧敏　硕士研究生，主研方向为数据库与信息管理系统。

给客户，进行方法调用和数据处理。

(2)客户端消息处理层

该层负责对相应消息和请求消息进行相应的处理。该层由一系列的消息处理器组成，每一组消息处理器都完成一定的功能，如加密/解密处理器，签名/验证处理器等。当然，该层提供了一种特殊的消息处理机制，用户还可以方便地自定义消息处理器，从而实现定制的商业逻辑。为了进行安全处理，该层必须从客户端获得一些附加消息，最为主要的就是用户的密钥。

(3)服务器消息处理层

服务器消息处理层除了处理安全性必需的解密/加密、验证/签名外，还需要提供用户数据库，以便进行用户的认证，以及访问控制策略的实施。此外，还可以对登录用户进行登记形成日志等。

(4)Web 服务层

该层主要由 SOAP 处理器和业务组件组成。SOAP 处理器负责将请求消息格式化对象调用消息，调用相应的业务组件，然后获得调用的结果，格式化为 XML 格式的消息，生成响应消息，经过安全处理层，发送给服务客户。

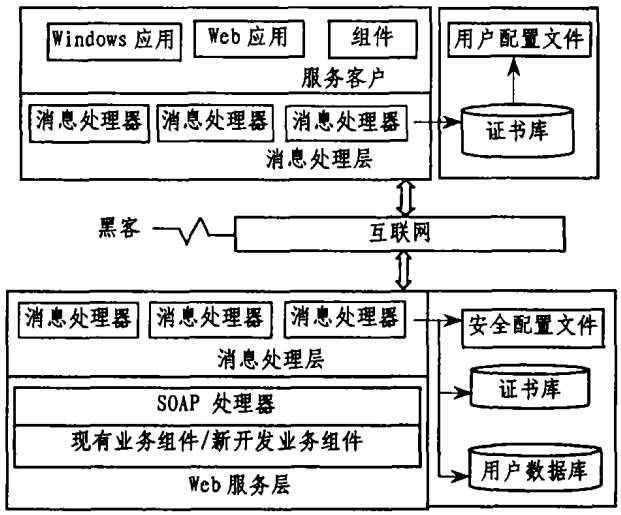

图1 基于消息层的 Web 服务安全模型

3.2 消息处理流程

图2给出了客户端从发出请求到接收响应后的消息处理流程。

消息处理的步骤为：

(1)客户端发送原始的对象调用信息，经过 SOAP 格式化程序串行化为 XML 文档。

(2)请求的消息经过消息处理层，相应的处理器提取系统参数，对消息进行处理。典型的情况是：加入用户的身份认证信息，对消息进行数字签名，而后进行加密处理，最后还可以进行压缩处理。经过这些处理后，消息继续传递给传输层。

(3)经过安全处理后的请求消息通过 Http 等传输层协议，发送给客户端。

(4)Web 服务器端接收到请求消息后，首先使用解压处理器解压缩消息，然后对用户进行认证，最后解密和验证数字签名。

(5)经过输入处理器集合处理的消息转化为了原始的请求消息，通过 SOAP 格式化程序转化为相应的对象调用。

(6)响应消息重复以上的步骤，传送给客户端。

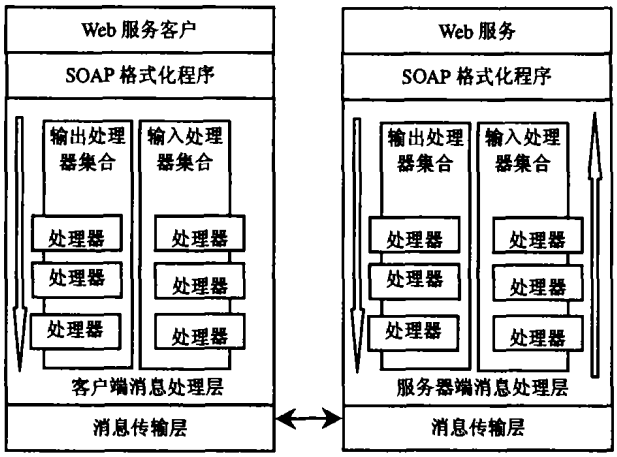

图2 Web 服务消息流程

4 Web 服务安全性模型的实现

4.1 系统实现的环境

本系统的实现环境采用 Micorosoft. NET，操作系统环境为 Windows 2000 Server，开发工具为 Visual Studio. NET 2003。. net 将 CryptoAPI 改编进 System. Security. Cryptography 名字空间，使密码服务摆脱了 SDK 平台的神秘性，变成了简单的. net 名字空间的使用。

4.2 Web 服务安全组件库的实现

4.2.1 *总体实现方案* 模型中处理器的实现以及 XML 安全规范的实现最终都封装在一个组件库中，物理形式为一个名为 MonicaSoft. WebService. dll 的文件，用户可以利用此文件进行安全 Web 服务的二次开发。该组件库的根名称空间为 MonicaSoft. WebServices。在这个名称空间里面，定义了基本的类和接口，用于支持二次开发。主要包括：安全性组件部署接口类、消息处理器模型类、消息处理层类和最终实现的安全消息综合处理器、访问控制处理器等。并定义了处理 XML 元素类需要定义的接口。在该名称空间下是 MonicaSoft. WebServices. Security 名称空间，在这个空间里面分别定了三个名称空间，即 Encryption、Signature 和 Token，分别代表了实现 XML 加密、XML 数字签名和实现安全性令牌的组件集合。这部分是整个 Web 服务安全组件库的核心代码。它具体实现了三个关于 XML 和 Web 服务安全的规范，即 XML 加密规范、XML 数字签名规范和 WS-Security 规范。

该组件库的实现主要包括安全性令牌、XML数据加密规范、XML数字签名、消息处理层以及访问控制策略的实现。由于篇幅所限，下面仅介绍消息处理层的实现方法。

4.2.2 *消息处理层的实现* 消息处理层的基本单元是消息处理器。消息处理器是处理消息的最小功能单元，其作用相当重要。消息处理器有两种类型：输入消息处理器和输出消息处理器。因此，定义了两个抽象类 SoapInputFilter 和 SoapOutputFilter。两个类中定义了消息处理的抽象方法 ProcessMessage。具体的消息处理器所需要做的就是在该算法内实现自定义的消息处理机制。例如，用户可以定义一个加密消息处理器 EncryptionOutputFilter。在它的 ProcessMessage 方法通过访问用户数据库，获得加密算法和加密密钥，并设置其它属性，调用 EncryptedData 对象的 Encrypt 方法。这样就完成了加密处理。消息处理器通过定义一个集合类来管理多种消息处理器。该类集成于 CollectionBase 类，并实现 ICloneable 接口。

消息输入处理层包含两个对象，即输出消息处理器集合和输入消息处理器集合。它拥有两个方法，即 ProcessInputMessage ( ) 和 ProcessOutputMessage()，分别用于处理输入消息和处理输出消息。

用户在使用 SOAP 协议调用 Web 服务方法的过程中，首先把方法调用序列化成 SOAP 消息，然后传送到服务器端，最后服务器端把 SOAP 消息反序列化成方法调用并执行相应的处理。Web 服务向客户返回的结果的过程与此类似。为了对 SOAP 消息进行扩展，.net 提供了 SoapExtension 类。通过继承这个类，用户可以在消息序列化和反序列化的前后添加自己的处理来改变 SOAP 消息，而且可以同时在客户端和服务器端添加这种扩展。通过这种方法可以解决消息处理层与 Web 服务的接口问题。

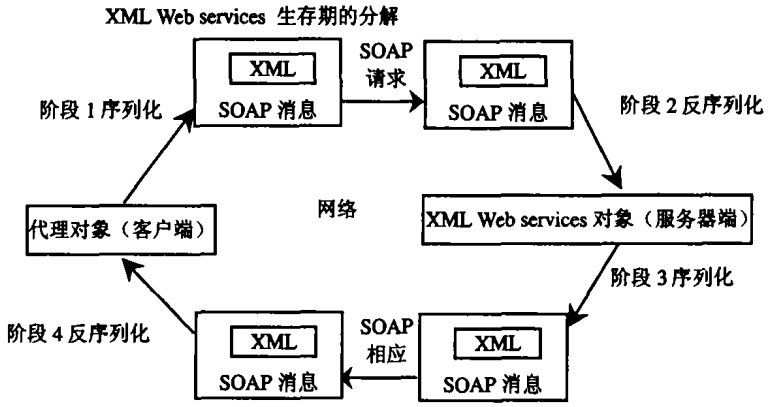

图3 Web 服务消息生存期图

SecurityExtension
-baseStream:Stream -newStream:MemoryStream -pipeline:Pipeline
-GetInitializer(in methodInfo: LogicalMethodInfo, in attribute:SoapExtensionAttribute):object -GetInitializer(in type:Type):object -Initialize(in initializer:object) -InitializePipeline() -ProcessMessage(in message:SoapMessage) -ChainStream(in stream:Stream):Stream -AfterDeserialize(in message:SoapMessage) -AfterSerialize(in message:SoapMessage) -AfterSerializeClient(in message:SoapMessage) -AfterSerializeServer(in message:SoapMessage) -BeforeSerialize(in message:SoapMessage) -BeforeDeSerialize(in message:SoapMessage) -BeforeDeSerializeServer(in message:SoapMessage) -BeforeDeSerializeClient(in message:SoapMessage)

图4 Web 服务接口类类图

图3给出了 Web 服务生存期的示意图。根据这个图，在消息不同的生存期通过截获 SOAP 消息，消息处理层对原始 SOAP 消息进行消息处理，就能达到透明地处理消息的目的。

根据 SoapExtension 类的定义，实现了 SecurityExtension 类。在这个类中，使用消息层对象，对消息进行处理。图4给出了该类的类图。

结束语 本文根据 W3C 以及微软、IBM 提出的 XML 加密规范，XML 数字签名规范和 WS-Security 等 XML 安全性规范以及访问控制等安全技术，提出了基于消息层的 Web 服务安全性模型，并在.net 平台实现了一个 Web 服务安全系统。通过分析和验证，该系统具有安全性、可扩展性、服务的透明性、安全的可控性、消息级端对端的安全性等诸多优点。该系统对于 Web 服务的安全具有重要的意义。

参考文献

1 Banerjee A, Corera A. C# Web 服务高级编程[M]. 北京：清华大学出版社，2002
2 朱玉，邓晓燕，昭培南. 基于 XML 的应用层安全解决方案[J]. 计算机工程，2003，29(2)：180～181，203
3 石伟鹏，杨小虎. 基于 SOAP 协议的 Web Service 的安全基础规范[J]. 计算机应用研究，2003，2：100～102，105
4 Basiura R, Conway R. Professional ASP.NET Security[M]. 美国：Wrox Press，2002
5 Microsoft. Security in a Web Services world: AProposed Architecture and Roadmap. http://www-900.ibm.com/developerWorks/cn/Webservices/ws-secmap/index_eng.shtml