基于协同的 web 服务安全模型构建及实现

Construction and Implementation of Web Services Security Model Based on Coordination

毛承品　范冰冰　龙 灿　（华南师范大学计算机学院 广州 510631）

摘要：首先分析 Web Service 安全性需求特点，当前针对 Web 服务的安全解决方案仅局限于局部解决方案，指出传统技术在解决 SOAP 消息传输中的缺陷以及 Web Service 安全的关键技术；基于这些安全技术，提出了新的 Web Service 体系安全模型，并实现 web 服务安全模型，最后将此模型用于互联网 Web Service 协同应用系统。

关键词：Web 服务　简单对象访问协议　web 服务安全　安全　协同

1　web service 安全问题引入

Web 服务就是一种部署在 Web 上的对象组件，通过一定的范例，开发人员只要建立一种标准化的接口，就可以通过 Internet 对各种各样的应用服务进行拼装和组合，从而形成新的应用服务。Web 服务具有不同于其它应用安全性需求的以下特点[1]：

元素级别的安全性：XML 封装的是一种结构化的数据，包含控制信息和消息内容，Web Service 中需要对同一文档的不同部分进行不同处理，即如何控制对不同元素组的授权查看。

异步多阶消息传递：SOAP 消息依赖于消息处理中介体转发消息的能力，特别是，SOAP 消息模型对抽象物理网络和应用程序基础架构的逻辑端点进行的操作。

端到端安全：在 Web Service 不仅要求消息发送端到中介点、中介点到中介点以及中介点到服务端消息安全，而且要求在整个消息发送的过程中，包括中介点上，它都是安全的，即要实现端对端的安全。

传输的独立性：Web Service 不规定传递消息的专用协议，它依靠和应用层协议的绑定来完成消息的传送任务，在传送消息的过程中可能会绑定不同的传输协议，所以当安全信息（如消息发送者的身份验证）需要被转移到消息路径上的下一个传输协议安全性域时，就可能会导致完整性方面的缺陷。

2　已有的消息解决方案及缺陷

SSL（Secure Sockets Layer，SSL）/TLS（Internet Protocol Security，IPSec）可用来保护任何位于 TCP 之上的应用层协议的通信安全，如 HTTP，FTP 和 TELNET。如果将传输层安全机制应用于 Web Services，则不能满足 Web Services 安全性的要求，原因如下：

首先，SSL 能保证相邻实体间点到点的消息安全，无法保证整个实体链端到端的消息安全；SSL 提供的是面向数据流的安全信道，无法实现对传输数据的细粒度安全保护。

第 2，SSL 目前仅和 HTTP 协议有绑定（HTTPS），不能为 SMTP 等其他传输协议提供安全保护，不具有传输独立性；SSL 能保证正在传输的数据的安全，保证存储后的数据的安全，无法提供 SOAP 消息长期确认性。

第 3，基于 SSL 的 HTTPS 在建立完共享密钥后，传递消息的时候并没有使用数字签名技术，因此不具有抗否认性的能力，而这在 web 服务中是不可或缺的。

第 4，HTTPS 不能根据不同的消息接受者，加密消息中的不同部分元素。

综上所述，在分布式应用和较强的互操作性环境中，传统的传输层安全机制虽然能够安全传输整体的数据信息，但在保证元素级数据安全方面却无能为力，同时它也不能实现端对端的安全性，另外，如果 Web 服务的调用跨越了使用不同安全性策略、不同的传输协议的不同的安全性域时，也需要在两个安全性域的界点上给出在转化安全性策略、更换传输协议时安全保障的解决方案。鉴于此，提出一个完整的处理机制，从整体上完善和解决好 Web service 安全问题。

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn

3　Web services 安全模型和模型安全性

3.1　Web Services 安全模型

鉴于前面提到的目前针对 Web service 解决方案的不足或缺点，本文提出和构建 web 服务安全模型（见图 1）；

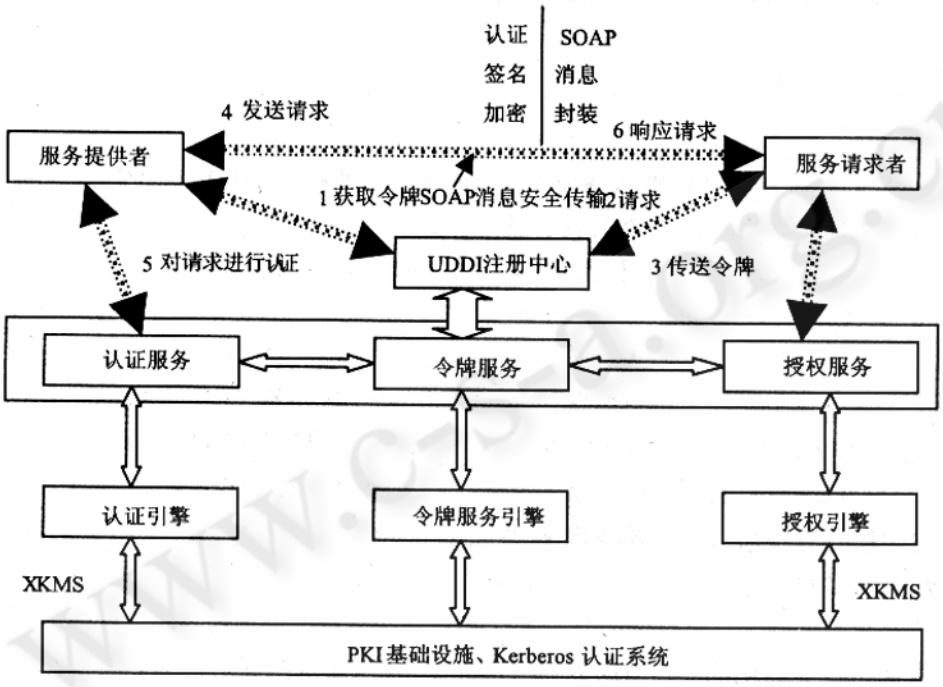

图 1　Web Service 安全模型结构图

Web service 安全模型的逻辑解释：服务提供者把 Web Services 的接口描述文件发布到 UDDI 注册中心，注册中心统一管理服务注册信息，服务请求者通过注册中心查询服务，查询到自己需要的服务后，需要把描述绑定到具体的提供者，然后调用；在这个过程中，请求者须获得相应的令牌及身份认证；同时 UDDI 需要对请求者进行认证，当请求者具有服务策略要求的声明时，才允许绑定并调用，如果服务提供者和请求者不在同一信任域，服务提供者须验证服务请求者的令牌，需要对令牌进行交换；

无论是认证、授权、或者是令牌交换，都是以独立服务的形式来提供，通过各自的引擎与 PKI 基础设施和 Kerberos 认证系统连接，获得基础设施提供的安全服务，完成认证、授权和令牌地交换功能。

3.2　Web Services 模型安全性

新提出的 Web Services 安全模型结构能够保证以下方面的安全：

消息的安全性：能保证消息的完整性、机密性、来源认证等，采用 XML 加密、XML 签名、令牌，WS－Security，SOAP 签名技术等。

用户身份的认证：用户需向认证、令牌服务发出请求，得到对用户身份的认证信息。这一部分的最底层使用 PKI 设施和 Kerberos 票据系统，它们通过认证引擎与认证服务器连接。

用户的授权\访问控制：用户可以向授权服务器发出请求，得到授权或者拒绝信息。这一部分的底层使用 PMI 设施，PMI 设施通过授权引擎与授权服务器连接。

联合信任：在不同信任域中的用户可以向令牌服务发出请求，令牌服务通过令牌服务引擎跟基础设施建立连接，获得服务方要求的令牌，从而建立信任。

本安全模型因考虑到整个 Web service 体系安全，提出针对 SOAP 、UDDI、WSDL 的具体安全解决方案：

（1）SOAP 安全的解决。将安全的基本要求（机密性、授权、数据完整性、原始性证明、不可抵赖性）扩展应用到整个的 SOAP 信息中，包括 SOAP 头以及 SOAP 体；其它安全措施也可结合 SOAP 层安全与传输层以及应用程序来共同解决。

SOAP 签名[1]：即消息身份验证、发送方/接收方身份验证以及不可抵赖性，将 XML 签名技术引入了 SOAP，形成了 SOAP 数字签名的标准。

SOAP 加密：SOAP 加密利用 XML 加密技术对 SOAP 进行扩展。

SOAP 认证：数字签名没有提供信息认证，恶意破坏者可以记录一个信息并反复发送（重复攻击），为了避免受这种类型的攻击，数字签名必须结合一定的方法来保证信息的唯一性，如：时间戳（Timestamps）等。

SOAP 授权：不同的用户享有不同的使用的权限，

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn

必须对服务使用者进行严格的授权管理，为了便于服务提供者对客户进行权限的管理，SOAP 头中包括了相应的权限信息和主题信息，进行 SOAP 授权的扩展。

（2）UDDI 安全的解决。UDDI 包括查询 API 和发布 API，查询 API 是面向公共匿名服务的，没有安全的限制；以下主要是针对发布 API 安全提出的解决方案。

用户发布服务前，需要获得 UDDI 服务器的授权，只有获取 UDDI 服务器认证和令牌的用户才能发布、修改、删除服务条目；用数字签名技术来保护注册数据的完整性和不可抵赖性；每一个服务条目可以定义自己的认证规则；对内部的条目的访问控制管理；需要对每一个请求进行身份验证和权限验证，保证端到端的安全性。

解决以上问题的关键在于认证和授权体制，每个注册中心都建立自己的令牌生成和认证机制。

（3）WSDL 安全的解决。服务发布者把 WSDL 文档发布出去，注册到相应的注册中心。Web 服务的使用者通过查询 UDDI，找到自己所需访问服务的 WSDL 文档描述，并根据其中的描述，生成相应的 SOAP 调用消息。WSDL 的安全性解决主要包括下面几个方面：

① WSDL 信息的传输安全性，参考上面讲到的 SOAP 消息的安全性。

② WSDL 信息不同部分对不同的请求者采取的不同的认证策略，通过引擎与 PKI 设施和 Kerberos 票据系统交互，得到不同用户认证信息。

③ 建立出错或失败处理机制；保证描述信息的完整性；需提供服务提供者的相关身份认证信息，此为强制身份认证的一部分。

由于 WSDL 文件本身也是 XML 文档，所以解决 WSDL 传输安全，信息完整性等问题，使用 XML 的加密和签名技术来解决。其它的修改特性、公开性等等问题通过认证和授权体制解决。

3.3 Web Services 安全模型实现

针对此模型，采用最新的 WS－Security 规范来实现，目前，很多的 Web 服务工具供应商包括 IBM Web Sphere 和 Microsoft．NET、BEA Web Logic 等都提供对 Ws－security 的实现。此模型主要采用 Microsoft．NET 开发平台，C#语言来实现，另外使用中 Web 服务的增强开发组件 Web Services Enhancements（WSE）2.0。

（1）安全策略文件[8]。Web 服务提供者基于 WS—Security 协议的安全机制定制自己的安全需求，如支持的安全令牌类型、加密算法、签名手段等。然后根据安全需求使用 WS—SecurityPolicy 生成相应的安全策略文件。最后，使用 WS—PolicyAttachment 将其与 Web 服务绑定。示例如下：

```
<PolicyDocument>
<Mappings>
<！一安全策略所关联的 Web 服务一>
<map to="http://www.yanbu.com/DemoService.asmx">
<default policy="#policy－GUID" />
</map>
</mappings>
<Policies>
<wsp:Policy WSU:Id="policy—GUID">
<wsse:Integrity wsp:Usage="wsp:Required''>
<wsse:TokenInfo>
<！一安全性令牌描述一>
<Security Token>
<！一使用 X.509 v3 证书进行签名一>
<wsse:Token Type>wsse:X509v3</wsse:Token Type>
</Security Token>
</wsse:TokenInfo>
</wsse:Integrity>
</wsp:Policy>
</policies>
</policyDocument>
```

（2）安全技术的实现

① 认证。示例采用 X.509 证书，考虑到 SOAP 能够穿越防火强，因此采用 BinarySecurityToken 令牌，BinarySecurityToken 类继承 Security Token 类，BinarySecurityToken 的标准格式：

```
<wsse: BinarySecurityToken wsu: Id=...
    Encoding Type=...
    Value Type=...>
    Binary Data ...
    <wsse: BinarySecurityToken/>
    Web 服务端：
    public class WSEBinarySecurityToken: BinarySecuri-
```


©《计算机系统应用》编辑部 http://www.c-s-a.org.cn

```
tyToken {
    GetSecurityToken(UToken)//获得用户的令牌
    GetSoapMessage();//获取 SOAP 消息
    Search(UToken){Validate();}//遍历用户令牌,
验证身份
    }
    }
```

② 消息加密。采用 X.509 证书加密,利用 Server 的证书加密消息时,采用标识式,通过标识定位到相应的证书;

```
<Wsse: SecurityTokenReference>
<EncryptedDataid? Type>
<! —加密算法的 URI—>
<Encryption Method/>
<! —密钥信息—>
<Key Info/>
<! —加密的数据内容—>
</Cipher Data>
</Cipher Value>
</Cipher Data>
<Encryption Properties>
</Encrypted Data>
</wsse: SecurityTokenReference>
```

代码示例使用 X.509 证书加/解密 SOAP 消息,忽略逻辑判断。

```
GetSoapMessage();//获取 SOAP 消息
GetCert();//获取证书 Cert
X509SecurityToken(Cert);//使用证书创建--个安全令牌
EncryptedData(Token);//加密消息
AddSoapMessage(Token,CipherValue,Key Info);//将
```

加密消息到 SOAP 消息的 Header 中

③ 消息签名。采用 X.509 签名,用户使用与自己证书对应的私钥签名消息的时,采用包含式的引用即同时将自己的证书传送给消息的接受方(Server),接收方利用证书的公钥验证签名。

```
<Wsse: SecurityTokenReference>
<Signature Id? >
<Signed Info>
<! —规范化法则的 URI—>
<CanonicalizationMethod/>
<! —数字签名算法—>
<SignautreMethod/>
<Reference/>
</Signed Info>
<Signature Value/>
<! —签署者公钥或 CA—>
<Key Info/>
</Signature>
</wsse: SecurityTokenReference>
```

忽略所有的逻辑判断,实现代码如下;

```
GetCert();//获得证书 Cert
X509SecurityToken("cert");//使用证书创建一个安全
令牌
GetSoapMessage();//获取 SOAP 消息
MessageSignature(Token);//签名消息
AddSoapMessage(Token,SignatureValue,Key Info);//
```

将加密消息到 SOAP 消息的 Header 中。

④ 其他安全组件:针对 SOAP 消息的输出端,进行时间戳过滤,防止重放攻击,置放于 SOAP 消息发送端;

```
GetSoapMessage();//SOAP 消息输出端,获取输出
SOAP 消息 env
TimestampOutputFilter();//定义时间戳输过滤器
ProcessMessageAdd Timestamp )//添加时间戳
```

SOAP 消息接收端;

```
TimestampInputFilter();//定义时间戳输入过滤器
ProcessMessage(env);//读取时间戳,若超期抛出异常
```

4 总结

本文提出了一种基于协同应用的 Web Service 安全模型,并阐述此模型能对 Web Service 提供的安全保证,以及在此模型上的安全通信机制,最重要的一点是,从 Web Service 整体角度提出 UDDI、SOAP、WSDL 的安全解决方案,从而形成安全的、可互操作的、实用的 Web 服务安全通信解决方案。此安全模型不仅适用于 Intranet 的内部 Web Service 环境,也适用于 Internet 环境下的 Web 服务协同应用系统,能够解决典型 Web 服务应用模式下的通信安全问题。此实例在 Intranet 环境的协同应用下进行实现和验证。不足之处,是没有在 Internet 协同应用环境下验证。

(下转第 81 页)

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn

(上接第 76 页)

参考文献

1 柴晓路、梁宇奇,Web Services 技术、架构、和应用,电子工业出版社,2003 年 1 月第一版.

2 Keijo Heljanko. Using Logic Programs with Stable Model Semantics to solve Deadlock and Reach ability Problems for 1 – safe Petri nets. www. tcs. hut. fi/ – kepa/publications.

3 M Gelfond, V Lifschitz. The Stable Model Semantics for Logic Programs[c]. In: Proceedings of the Fifth International symposium on Logic Programming, Cambridge. Ma, MIT Press,1988:1070 – 1080 3.

4 DEastlake, JReagle1XML2signaturesyntaxandproce – ssing1Amsterdam: W3C, 20021, http://www. w3. org/TR/xmldsig2core.

5 PHallam2Bake1XMLkeymanagementspecification (XKMS) Version2101Amsterdam: W3C, 20031 http://www. w3. orgTR/xkms2/.

6 PHallam2Baker, EMaler1Securityassertionmarkuplanguag (SAML), Version1101Billerica, MA: OASIS, 20021 http:/www. oasis2open. org/committees/security/docs/.

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn